of a message. In a contention bus or contention ring network, the output machine may transmit only when the network is quiet. The "token present" signal is replaced by a "network quiet" signal. In the ring network, the reception control section signals the transmission control section if it detects another token in the midst of its receipt of the message the transmission control section sent; this has its analogue in the collision detection capability of the contention network. In both cases, the LNI must abort transmission of its message and take corrective action. In the ring network this is an error condition, an exception; more than one control token is present in the ring. In the contention network, a collision is an expected event. Both situations can be handled by the LNI reporting the event to host software, which can attempt to restart a token on the ring, in the ring network case, or apply a retransmission backoff algorithm in the contention network case.

A better solution for the contention network is to modify the transmission control section to execute a simple retransmission backoff algorithm in hardware. This requires that the entire message remain accessible to the transmission control section without host intervention. The FIFO buffer cannot be used in this situation; a complete packet buffer which is not erased until the message has been successfully transmitted is an appropriate alternative.

Two features of the ring nctwork LNI's transmission control section are not needed in the contention bus network version: the data repeater which passes bits from the receive side of the LNI to its transmit side when the LNI is not transmitting a message, and the token gencrator which places a new token or connector onto a quiescent ring. Of course, the connector is a brief sequence of bits, and there is no good motivation to delete it from the beginning of messages transmitted by the contention bus version of the LNI. In fact, retention of the connector at the head of a message results in fewer changes to the input machine of the LNI. It can use its token/connector detector to signal the beginning of an incoming message. Its function remains the same, for the most part; extra connectors detected in the middle of a message indicate a collision, just as they do for the ring network version. However, in the contention bus network, because bits are not repeated from one LNI to another, there is no way to set the match/accept bits for the benefit of the transmitting LNI, and the match/accept field of the message cannot be used.

The signal conditioning section of the LNI undergoes an interesting transformation. For a contention ring network, of course, the signal conditioning section remains the same. However, for a contention bus network, the logic levels of the LNI must be converted to appropriate signal levels and waveforms for the coaxial cable of the bus. This is done in a two-step process. First, a cable transceiver is added to the configuration. To minimize impedence mismatches, reflections, etc., the transceiver is located immediately adjacent to the network cable, and is often packaged separately from the LNI.[4] It is connected to the cable either directly, or via a short stub cable attached to the main cable via a tap. Second, since the transceiver is located adjacent to the network bus cable, and the LNI is located next to its host, an appropriate transmission scheme must be selected to span the intervening distance. For distances up to 30 ft or so, "single-ended" drivers and receivers will suffice. For better reliability, greater distances, or both, differential signals over a shielded twisted pair can be used—just as in the transmission medium of the ring network itself. So, the signal conditioning section of the original LNI can be modified to interconnect the LNI and the cable transceiver.

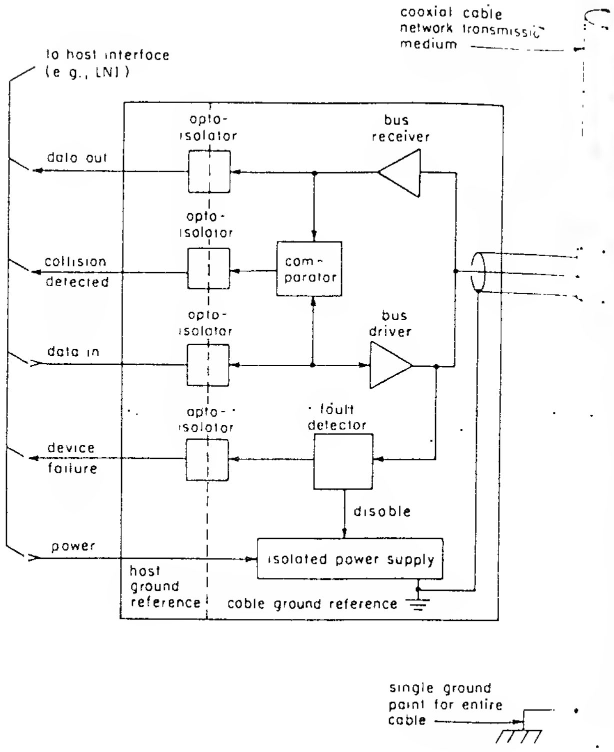


Fig. 8. A typical bus transceiver. The opto-isolators and isolated power supply permit the drivers and receivers to be referenced to cable ground; the cable, in turn, is grounded at only one point along its length, eliminating problems that would result if each transceiver tied the cable to local host ground.

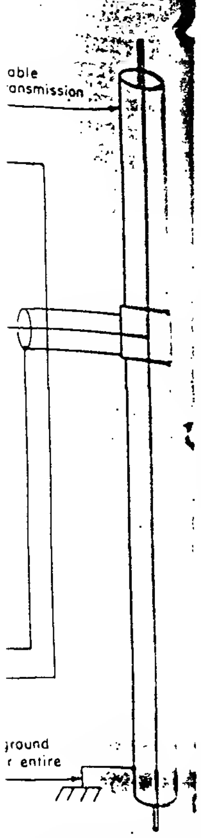
4) The Cable Transceiver: The care taken in the design of a cable transceiver for a contention bus network will strongly influence the overall reliability and performance of the network. Therefore, we conclude our case study by examining a hypothetical contention bus cable transceiver, shown in Fig. 8, that is similar to one designed and built for the CHAOS Network at the MIT Artificial Intelligence Laboratory; it is typical of transceivers built for various contention bus networks.

The cable transceiver performs the following functions:

1) transmission (cable driving);
2) reception;
3) power and ground isolation;
4) collision detection;
5) transceiver fault detection ("watchdog").

The first three of these constitute part of the signal conditioning function described previously.

[4] This has become common practice in local area networking; the networking transmission medium is generally *not* brought into the racks, equipment bays, etc., of a host computer where it would be subject to accidental disconnection and other physical abuse that could disrupt the entire network. Instead, the connection point for a host is designed to be physically stable: a box on the wall, above a false ceiling, etc.



rs and isolated powe
referenced to cabl
y one point along th
f each transceiver the

ible via a tap. Sev-
ent to the network
its host, an appro
ected to span the
30 ft or so, "single
or better reliability.
nals over a shielded
ansmission medium
conditioning section
iterconnect the LNI

aken in the design
bus network will
y and performance
our case study by
is cable transceiver.
esigned and built for
Intelligence Labora
r various contention

wing functions:

og").

of the signal cond

The basic design principle of the transceiver is that it must present a high impedence to the bus except when it is transmitting and actually driving the bus. This is essential to the operation of the contention bus network; a large number of receivers on the bus must not present impedence lumps or in any way interfere with a transceiver which is actively transmitting.

The receiver must be able to detect and properly receive signals from the most distant point on the bus; in addition, it must be able to detect a colliding signal while its companion transmitter is itself driving the bus. This requirement impacts the choice of an encoding scheme for data transmitted on the bus. A number of data encoding schemes can be used, all of which require that the transmitter be able to place the transmission medium in two distinct states. At first glance, it might seem that *three* states could be used: the quiescent, high-impedence state, to indicate that no transmission is in progress, and two active driver states, for example $+V$ and $-V$. However, with two active driver states, when two or more network nodes attempt to transmit simultaneously, the cable will be driven to different voltage levels at different points. This has two effects. First, it places a severe load on drivers. Second, it makes the detection of a colliding signal more difficult than it needs to be. On the other hand, if the transceiver drives the cable to some voltage to represent one signaling state, and represents the other signaling state by *not* driving the cable, the problem of overloaded drivers is eliminated, and the task of collision detection is greatly simplified. Collision detection is accomplished looking at the bus during the transmitter's quiescent state. Any signal present during that time must come from another transceiver, and constitutes a collision. The transceiver can detect an incoming signal with 20-dB attenuation, which corresponds to about 1 km of the particular cable used.

The transceiver must be able to cope with ground potential differences at the various network hosts. Isolation is accomplished by high-speed optocouplers and an isolated power supply which enables the major circuit elements of the transceiver to be referenced to cable ground, rather than local host ground. Finally, the fault detection, or watchdog circuit examines the output of the driver to guard against transceiver failures which drive the bus and disrupt the network. The signaling states used by the transceiver result in the driver being quiescent approximately 50 percent of the time; if the driver remains on steadily for several bit-times, it is deemed to be faulty, and the fault detector disconnects its power, which, of course, returns the driver to its high-impedence state.

5) *Complexity of the Local Network Interface:* In its present form, the LNI comprises about 350 TTL SSI and MSI integrated circuits, apportioned as follows:

PDP-11 full-duplex DMA	100
Name table controller	25
Name table cells (8 provided)	90
Network-oriented portion	120
Test and diagnostic	15
Total	350

The count of 120 chips for the network-oriented portion of the LNI, excluding the associative name table, is well within the capabilities of current large-scale integration. As the field of local area networking matures, and standards are arrived at, it is likely that integrated circuit manufacturers will add local area network controllers to their product lines, to take their place alongside other LSI data communication chips which are already available, making high-performance local area network technology available at a very reasonable cost.

V. Protocols for Local Area Networks

As in long-haul networks, local area network protocols can be divided into two basic levels—low-level protocols and high-level protocols. At each level, the characteristics of local networks impact effects on protocol design and functionality.

A. Low-Level Protocols

The term *low-level* protocol identifies the basic protocols used to transport groups of bits through the network with appropriate timeliness and reliability. The low-level protocols are not aware of the meaning of the bits being transported, as distinct from higher level application protocols that use the bits to communicate about remote actions. Two aspects of local area networks have a very strong impact upon the design of low-level protocols. First, the high performance achievable purely through hardware technology enables the simplification of protocols. Second, low-level protocols must be designed to take advantage of and preserve the special capabilities of local networks, so that these capabilities can be utilized, in turn, by higher level application protocols. We will explore these two issues in this section.

1) Simplicity: Local area networks must support a wide variety of hosts, from dedicated microprocessors to large time-sharing systems. The existence of extremely simple hosts (such as microprocessor-based intelligent terminals, or even microprocessor printer controllers) leads to a desire for simple, flexible, low-level protocols that can be economically implemented on small hosts, while not compromising the performance of large hosts. Supporting a variety of hosts also leads to a difficult software production and maintenance problem that can be ameliorated somewhat by having a protocol that is simple to implement for each new kind of host. Although quite a variety of hosts has been attached to long-haul networks such as the ARPANET, the problem of software development has not been too severe, since each individual host in such environments usually has a software maintenance and development staff. In the local area network context where a variety of computers are all maintained by a small programming staff, the arguments for simplicity in protocol design are far stronger in our view.

In a long-haul network, complexity results from strategies that attempt to make as much of the costly network bandwidth as possible available for transport of high-level data. The costs of a local area network are concentrated instead in the host interfaces, the hosts themselves, and their software. Two factors lead to the simplicity of low-level local area network protocols.

a) Unrestricted use of overhead bits: Bandwidth is inexpensive in a local area network; there is little motivation to be concerned with protocol features designed to reduce the size of the header or overhead bits sent with each message. This is in contrast to protocols developed for networks making the more conventional assumption that bandwidth is expensive. For example, the ARPANET NCP host-to-host protocol [26] initiates a connection using a 56-bit (net, host, socket) identifier for the destination, but then goes through a negotiation so that instead of sending this 56-bit value on subsequent messages, a 32-bit (net, host, link) value can be sent instead. It is not clear whether this conservation of bits is appropriate even in a long-haul network; in a local area network, where bandwidth is inexpensive, it is clearly irrelevant. Other examples of ways in which extra header space can be used to simplify processing include:

1) having a single standard header format with fields in fixed locations, rather than having optional fields or multiple packet types; field extraction at the host can be optimized, reducing processing time;
2) using addresses that directly translate into addresses of queues, buffers, ports, or processes at the receiver without table lookup.

b) Simplified flow control, etc.: The low transmission delay inherent in local area networks, as well as their high data rate, can eliminate the need for complex buffer management, flow control, and network congestion control mechanisms. Consider, for example, flow control: the problem of assuring that messages arrive at the recipient at the rate it can handle, neither too fast, so that its buffers overflow, nor too slow, so that it must wait for the next message after processing the previous one. In a long-haul network, a receiver typically allocates to the transmitter enough buffer space for several messages following the one currently processed by the receiver, so that messages can be placed in transit well before the receiver is ready to process them. Considerable mechanism is required to keep the sender and the receiver properly synchronized under these circumstances. In a local area network, the delay will typically be low enough for a much simpler flow control mechanism to be employed. For example, one can use the very simple strategy of not sending a message until the recipient has explicitly indicated, by a message in the other direction, that it is ready for it. In contrast, a network using communication satellites has such a high transmission delay that very complex predictive flow control algorithms must be used to obtain reasonable data throughput.

It is crucial to understand that other factors may obviate these simplifications. While the data rate and delay characteristics of a local area network can render it essentially instantaneous, its speed cannot eliminate the intrinsic disparity that may exist between the capabilities of two hosts that wish to communicate with each other. These disparities may not show up when the two hosts are communicating through a long-haul network whose characteristics are so constraining that the principal problem is dealing with the restrictions of the network. While protocols for local area networks need not include mechanisms designed to cope with the limitations of the network itself, it is still necessary to design protocols with sufficient generality to cope with disparities between the capabilities of machines wishing to communicate through the network. Such disparities include:

1) mismatch between the rate at which hosts can generate and absorb data;
2) host delay between the time a packet is received and the time it is successfully processed and acknowledged:
3) amount of buffer space available at the sender and the receiver.

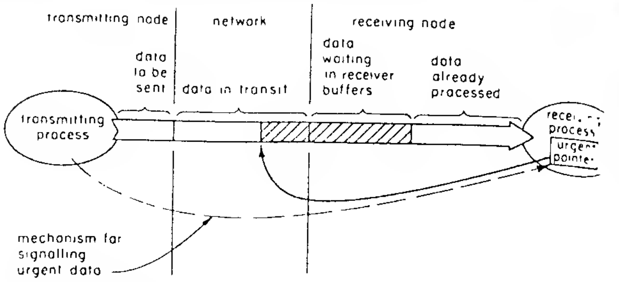


Fig. 9. The urgent pointer mechanism. By transmitting a new, larger value of the urgent pointer, a pointer into the data stream, a sender can indicate the data bufferred in the sender, network, and receiver are holding up data that must be processed quickly. The receiver can then adjust his use of the data stream flow control to process the bufferred data until the urgent data is processed. The shaded area indicates the location of potentially urgent data specified by a particular urgent pointer value.

Further, considerable effort may be required to modify host software to provide a suitable interface to the network. If one were to consider the simple flow control mechanism mentioned earlier, where a message is sent in the reverse direction requesting transmission of each message as it is needed, one would discover that in many cases the scheme was unworkable, not because the network introduced intolerable delays, but because the hosts communicating with each other themselves introduced excessive delay. In a large host with a time-shared operating system, for example, the real time that elapses from the time a message is received, one or more processes are scheduled in response to this message, and that process runs, to the time a message is sent in response, could well run into a large number of milliseconds milliseconds during which the other host is forced to wait.

c) Example of protocol simplification: The low-level protocol initially proposed for the Laboratory for Computer Science Network at MIT is an example of the sort of protocol that results when simplicity of mechanism is a primary design goal. The Data Stream Protocol (DSP) was based on the Transmission Control Protocol (TCP) used in internetworking experiments sponsored by the Defense Advanced Research Projects Agency [27], but evolved from original TCP due to the continuing desire to simplify the protocol features, packet formats, and implementation strategies. Most of these simplifications have subsequently been incorporated into the TCP

One specific example is the mechanism used to signal *interrupts* and other urgent messages that are logically part of the sequence of data in a virtual circuit. The basic model is that the sender occasionally wants to signal the receiver that all data in the stream preceding the signal (buffered somewhere in the network) must be scanned immediately in order to respond promptly to some other important signal. A mechanism is provided whereby a pointer into the data stream is maintained at the receiver, which can be moved, when the sender chooses, to point to a more recently transmitted piece of data. This pointer, called the *urgent pointer*, can be used to indicate the point in the data stream beyond which there is no more urgent data. (See Fig. 9.) The urgent pointer can be implemented in two ways, depending upon the nature of the host receiving the message. In the case of a simple (e.g., microprocessor) host dedicated to a task that processes the incoming stream as it arrives, the host need not process the urgent pointer, since by design, all data, urgent or not, are processed as quickly as possible. In contrast, on a large time-shared host, data need not be processed until either

a) the process to receive the data is scheduled and requests input, or b) the urgent pointer points to data not already received by the process. In case b) an interrupt is sent to the receiving process, indicating that data should be read and processed until the urgent pointer is past. The corresponding mechanism in TCP required that a host be capable of understanding and responding to a special interrupt signal in the data stream, even if the signal had no meaning to the host in its particular application of TCP. The urgent pointer, then, is a simple mechanism that meets the needs of sophisticated host implementations without placing an excessive burden on unsophisticated hosts.

2) *Special Capabilities:* The other aspect of low-level protocols for local area networks to be discussed is the manner in which protocols must be structured to take advantage of, and provide to higher levels, the unique capabilities of local networks. Conventional low-level protocols have provided a function best characterized as a bidirectional stream of bits between two communicating entities—a *virtual circuit.* The virtual circuit is implemented by a process that provides sequenced delivery of packets at the destination. While a virtual circuit is one important form of communication, two others easily provided by a local network are very useful in a variety of contexts. These are *message exchange* communication, where the packets exchanged are not viewed as being members of a sequence of packets but are rather isolated exchanges, and *broadcast* communication in which messages are sent not to one particular recipient but to a selected subset of the potential recipients on the network.

a) Message exchange: A typical example of a message exchange is the situation in which one message asks a question and another provides the answer. For example, if there are a large number of services provided by nodes connected to a local net, it is disadvantageous to maintain, on every node, a table giving all of the addresses of these, for whenever a change is made in the network address of any service, every node's table will need to be revised. Rather, it may be advantageous to maintain, as a network service, a facility which will take the name of a desired entity and give back its network address. Clearly, the pattern of communication with this service is not one of opening a connection and exchanging a large number of messages, but instead is a simple two-message exchange, with a query of the form "What is the address of such and such a service?" and a reply of similarly simple form. While a virtual circuit *could* be used for this exchange, it is unneeded and uses excessive resources.

b) Broadcast: The example given above demonstrates the need for a broadcast mechanism. If the service described above is intended to provide the address of network services, how can we find the address of this service itself? An obvious solution is to broadcast the request for information. The query then takes the form "Would anyone who knows the address of such and such a network service please send it to me?" There are many other examples, some apparently trivial but nonetheless very useful, for support of broadcast queries on a local network. A microprocessor with no calendar clock may broadcast a request for the time of day. A new host attached to the network for the first time may broadcast a message announcing its presence, so that those who maintain tables may discover its existence and record the fact. Broadcast mechanisms in the low-level protocols can also be quite useful in implementing higher level protocols for such applications as document distribution to multiple host nodes, and for speech and video conference calls.

Why are these alternative models of communication not commonly found in traditional networks? The first, and perhaps most important reason is that long-haul networks have not been extensively exploited for applications in which computers directly query other computers with individual, self-contained queries. Instead, the major use of long-haul networks has been for long-term, human-initiated interactions with computers, such as direct terminal use of a remote computer, or long-term attachments of remote job entry stations. Such human interactions usually involve many message exchanges between sender and receiver, so that the extra delay and cost of initial setup of a virtual circuit is insignificant—perhaps even recovered by reducing redundant information in each message. As new applications such as distributed data base systems become more important, these alternative models will become important in long-haul networks, but long-lived connections between terminals and host computers continue to dominate the usage.

The second reason is precisely that discussed in the previous section concerning the relative simplicity of protocols for local area networks—a variety of functions performed in conventional networks are very difficult to understand except in the context of a sequence of ordered messages (a virtual circuit) exchanged between two nodes. For example, flow control is normally handled in network protocols by placing an upper bound on the number of messages which may be flowing at any one time between the sender and the receiver. This concept has meaning only in the restricted case where the sender and the receiver are a well-identified pair exchanging a sequence of messages. There is no obvious equivalent of flow control that can be applied to situations where sender and receiver communicate by sending arbitrary unsequenced messages, or where a sender broadcasts to several receivers. Similarly, if efficiency requires use of the shorthand version of an address for communication between the sender and the receiver, this clearly implies that the sender and the receiver have negotiated this address, and agree to use it over some sequence of messages. Again, this idea makes no sense if communication is isolated in unsequenced messages.

Another problem that is traditionally handled in the context of a sequence of messages is the acknowledgment to the sender that the receiver has correctly received a message. If messages are sequenced, acknowledgment can be very easily done by acknowledging the highest member of the sequence that has been successfully received. If messages bear no relationship to each other, then each must be identified uniquely by the sender, and acknowledged uniquely by the receiver. This increases the complexity and overhead of acknowledgment. However, in most cases where message exchange communication is the appropriate underlying communication model, no acknowledgment mechanism is required of the low-level protocol at all. For example, if a microprocessor system asks the time of day, it is not at all necessary to acknowledge that the query has been successfully received; the receipt of the correct time is sufficient acknowledgment. Similarly, a request for a network address is acknowledged by a return message that contains the desired address. Depending on a low-level acknowledgment message to handle all failures can be dangerous, for it may lead to the practice of assuming that acknowledgment of receipt of a message implies that the message was processed at a high level.

In the broadcast context, it is difficult to formulate a useful definition of acknowledgment that can be supported by a low-level protocol. What does it mean to say that a broadcast message has been successfully received? By one of the possible recipients? By all of the possible recipients? One appropriate strategy is to rely on the high-level application to deal with these problems as a part of its normal operation, rather than have the low-level protocol concern itself with issues of flow control or acknowledgment at all.

3) Protocol Structure: Based on the previous observations, a two-layer structure is a very natural one for low-level protocols in a local area network. The bottom layer should provide the basic function of delivering an addressed message to its (one or many) destinations. This level corresponds to the concept of a *datagram* network [28]. It should also take on the responsibility of detecting that a message has been damaged in transit. To this end it may append a checksum to a message and verify the checksum on receipt. However, this layer probably should not take on the responsibility of ensuring that messages are delivered, and delivered in the order sent, since different applications have different needs and requirements for these functions. The first layer might be implemented entirely in hardware; however, if the packet size, addressing structure, or routing topology of the hardware is not sufficient to provide adequate message size, process addressing, or broadcast selectivity, some software help will be needed to make up the difference.

Above this first layer should be made available a variety of protocols. One protocol should support a virtual circuit mechanism, since a virtual circuit is definitely the appropriate model for a great deal of the communication that will go on in any network, local or otherwise. As alternatives to the virtual circuit protocol, there should be mechanisms for sending isolated messages, for message exchange communication, and additional alternatives to provide support for message models other than the ones we have discussed here. For example, transmission of digitized speech requires a communication model with some but not all of the attributes of the virtual circuit; in particular, reliability is of less concern than timeliness of arrival.

B. Applications of Local Area Networks; Higher Level Protocols

In the previous section we considered low-level protocols for a local area network. These protocols exist, of course, to support higher level protocols, which, in turn, support user applications. In this section we will consider a number of applications for which local area networks are suited.

1) Access to Common Resources: The model of computing most common over the last few years is that of a large centralized computer, with the only remote components being terminals and, perhaps, a few other I/O devices. Line control protocols such as SDLC [19] were created to serve this sort of arrangement. A simple but very important application of a local area network is to generalize this picture very slightly to include more than one central computer. As the total workload grows to exceed the capacity of a single machine, a common solution is to procure a second machine, and to divide the applications and workload between the two. The communication problem to be solved in this arrangement is simple but critical—to allow an individual terminal to have access to both of the central machines. A local area network can solve this problem, and provide some additional capabilities as well. For example, if the central facility has specialized I/O devices such as plotters or microfilm writers, they can be placed on the local area network and made accessible to both central machines—an advantage if a device is expensive and is not heavily enough loaded to justify having one for each computer. Further, I/O devices can be placed remote from the central site but convenient to users; for example a line printer can be placed near a cluster of users.

This pattern of sharing among several computers can be expanded to include more than just I/O devices. In fact the network can be used to move computations from one machine to another in order to spread the computing load equally. The high speeds available in the local area networks make this sort of load leveling much more practical than do the bandwidths traditionally available on long-haul networks.

2) Decentralized Computing: A wide variety of new uses for a local area network arises if the computing power available is not strongly centralized. Let us consider the alternative of a computing environment consisting of a large number of relatively small machines, each dedicated to a small number of users or a small number of tasks. In the extreme, we can look to the future and imagine the day when each user has a computer on his desk instead of a terminal. Such a completely distributed computing environment by no means eliminates the need for an interconnecting network, for users will still need to exchange information. Data files containing the results of one person's computation will need to be shipped through the local area network to be used as input to other tasks. Users will wish to communicate with each other by exchanging computer mail, as is now done over the ARPANET [29]. Users will still want access to specialized resources which cannot be provided to each user: resources such as large archival storage systems, specialized output devices such as photo typesetters, or connection points to long-haul networks. All of these features can be made available through the local area network.

3) Protocol and Operating System Support: The applications outlined in the previous paragraph can be supported by high-level protocols very similar to the ones already in existence in the ARPANET: TELNET for logging into a remote system through the network, and File Transfer Protocol for exchanging data between machines [26]. When one examines how these protocols might be modified to take advantage of the special attributes of a local area network, for example, its higher speed, one discovers that the problem is not one of modifying the protocols, but of modifying the operating system of the hosts connected to the network so that the services available through the network appear to be a natural part of the programming environment of the operating system. The File Transfer Protocol in the ARPANET, for example, is usually made available to the user as an explicit command which he may invoke to move a file from one machine to another. As part of this invocation he may be required to identify himself at the other machine, and give explicit file names in the syntax of the local and the foreign machine, describing exactly what action he wishes to perform.

This particular view of file transfer has two disadvantages. First, there is a lot of overhead associated with moving a file. Much of the delay in moving the file seen by the user has nothing to do with the time required to send the data itself through the network, but is rather the time spent establishing the connection, identifying the user at the other site, etc. Second, the file system on the local computer understands nothing about the existence of files accessible through the network. No matter how sophisticated the local file system in terms of keeping track of the various files that the user cares about, it requires explicit user intervention in order to reach through the network and retrieve a file from another machine. The use of a high-speed local area network will not eliminate any of these problems, but will instead make even more obvious to the user the overhead that the protocol imposes on the transfer of data. Clearly, what is needed is a further integration of the local area network into the file system and user authentication mechanism of the individual operating systems, so that interchange of information between the various machines can he done with less direct user intervention. Some attempts have been made to do this within the context of the ARPANET. RSEXEC is an example of a protocol which makes files on various TENEX operating systems in the ARPANET appear to the user to exist on a single machine [30].

The design of operating system structures to take full advantage of the capabilities of local area networks represents the current edge of research in this area. Examples of operating systems that incorporate a high-speed local area network into their architecture are the Distributed Computing System [31], the Distributed Loop Operating System [11], and MININET [32].

VI. Interconnection of Local Area Networks with other Networks

A. Motivation for Interconnection

As was mentioned earlier, a local area network will be only a part of the overall communication system used by the hosts attached to it. A very important use of the local area network can be to provide an interconnection between hosts attached to a local area network and other networks such as long-haul packet-switched networks and point-to-point transmission links. The advantage of this method of interconnection is reduced cost, by taking advantage of the fact that connection of a host to a local area network is relatively inexpensive. Instead of connecting all machines directly to the long-haul network, one can connect all the host computers to the local area network, with one machine, the *gateway*, connected to both the local area network and the long-haul network.

B. Protocol Compatibility

There are two pitfalls that should be avoided when planning for the interconnection of a local area network with a long-haul network. On the one hand, long-haul networks currently cannot provide all of the functions that local area networks can. If a local area network is initially designed to serve only the function of connecting hosts to a long-haul network, the protocols of the local network may be designed to serve only the needs of communicating with the long-haul network, and may not support the other functions that make a local area network especially attractive. On the other hand, if a local network is initially designed with no thought given to the possibility that it may be interconnected with another network, the protocols designed for it may lack the necessary generality. For example, the addressing structure used on the local area network may not be able to express destinations outside the local network. In either case, the only after-the-fact solution is to implement a second set of protocols for the local area network, so that different protocols are used for intercommunication with long-haul networks and for local services. This proliferation of protocols is undesirable, as it adds to the cost of software development associated with each new host added to the local area network. To avoid these pitfalls, it is important that all the functions a local area network is to provide must be considered from the very inception of the design of the network, and the protocols for the network must be designed to support that entire range of functionality.

Fortunately, initial experiments with protocols for local area networks suggest that a uniform approach to protocol design can support both specialized local network functions and interconnection with other networks, provided that both functions are envisioned from the start. Although the protocols used in the local area network must be made slightly more general to handle the internetworking situation, there is no interference with the realization of the purely local network functions. For example, a more general address field must be used to specify the destination of a message, but the only overhead implied if this same addressing structure is used for purely local messages is additional bits in the message to hold a presumably larger address. Since bandwidth is inexpensive, the bits "wasted" on this larger address are presumably irrelevant.

A slightly more difficult problem, one that is still being studied, is the problem of speed matching between the local area network and the long-haul network. As this paper has characterized the difference between local nets and long-haul nets, it is reasonable to presume that the local network will have a much higher data rate. If a host sends a large number of packets into the local area network with an ultimate destination to be reached through the long-haul network, the packets may arrive at the gateway much faster then the gateway can pass them to the long-haul network. Some mechanism will be required to prevent the gateway from exhausting its buffer space. The speed matching problem is not unique to the gateway between the local area network and the long-haul network; it occurs any time two networks of differing speed are connected together. (The problem may be more extreme here, though, due to the greater speed difference that can be encountered between local area and some long-haul networks. Satellite networks with speeds comparible to local networks are quite conceivable, yet are a long-haul technology.) A general discussion of the problems of internetworking, and some proposed solutions can be found in a companion paper by Cerf and Kirstein in this issue [33].

At the next higher level of protocol, one finds facilities that support various communications models, such as virtual circuits, broadcast, and message exchange. In interconnecting to a long-haul network we are chiefly forced to deal with a virtual circuit model, since that is the only pattern of communication usually supported by commercial long-haul networks. Here, it is appropriate to use a virtual circuit protocol in the local area network as similar as possible to that used in the long-haul network, so that translation between the two is easy. Although there is not as much practical experience available in the area of network interconnection as could be desired, it appears that one can develop a virtual circuit protocol for a local area network that is a compatible subset (in the sense of using compatible packet formats and control algorithms) of a suitable long-haul virtual circuit protocol. This means that it is not necessary to implement two complete virtual circuit protocols, one for internal local network use and the other for communication out through the local net. It leaves unanswered the question of how the additional features, such as complex flow control, buffering, and speed matching required for the long-haul protocol should be implemented. One approach would be to implement them in every host that desires to communicate over the long-haul network; this implies a programming burden for every machine. An alternative would be to implement the additional functions in the gateway machine that interconnects the local area network to the long-haul network. This would add considerable complexity to the gateway, for it will have to cope with such problems as the speed differential between the two networks without having the benefit of the flow control mechanisms normally used for this purpose in the long-haul network. At this time, it is not clear whether the gateway can assume the entire responsibility for augmenting a local network virtual circuit protocol with the functions required for communication through a long-haul network.

It would be advantageous to make sure the local area network protocols are also compatible with other communication models, such as single message exchange or selective broadcast, that may become available on commercial long-haul networks in the future. However, this presupposes that the long-haul networks attached to the local area network use a two-layer low-level protocol implementation such as that described for the local area network, and if the long-haul networks do use such an implementation, that they provide an interface that allows direct use of the datagram layer. Many current long-haul networks do not provide that interface.

VII. The Subnetwork Concept

Resting midway between the monolithic, single-technology, local area network and the internetworking environment is an approach to local area networking that we term the *subnetwork concept*, which provides for a mix of network technologies within a uniform addressing and administrative structure.

A. General Approach

A local area network can be composed of a collection of subnetworks, possibly implemented with various network technologies and perhaps with various transmission rates, but using identical software protocols, compatible packet sizes, and a single overall homogeneous address space.[5] These subnetworks are interconnected by *bridges*, which are midway in complexity between the repeaters used in a multisegment contention bus network (ETHERNET) and the gateway processor used between networks in an internetworking environment. This general structure is indicated in Fig. 10. A bridge links two subnetworks, generally at a location at which they are physically adjacent, and selectively repeats packets from each of them to the other, according to a "filter function."[6] In addition, since they buffer the packets they repeat, they can perform a speed-matching function as well.

B. Advantages of Subnetworking

The subnetworking concept enables a variety of technologies and data rates to be utilized in a single local area network, each to its best advantage. For example, a network could be constructed with a contention bus subnetwork, perhaps using coaxial cable originally installed for CATV, and with a ring subnetwork, using twisted pair which can be easily installed in a crowded laboratory environment. These two subnetworks could be of different data rates; the bridge between the two will handle the speed difference between them.

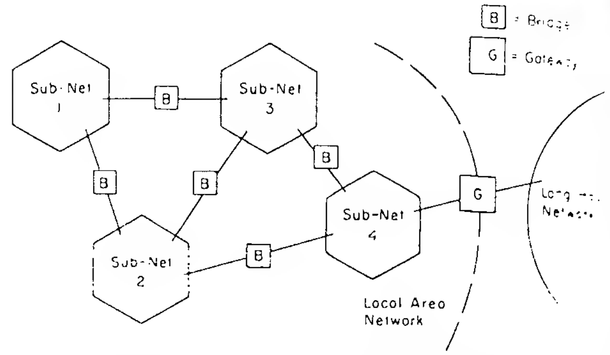


Fig. 10. The subnetwork concept. Here, a local area network is composed of a number of subnetworks, linked in some fashion by bridges. The subnetworks, though of differing technologies, share one address space, and the same protocols are used over the entire network. Thus the bridges can be simpler than the gateway which connects the local area network to the long-haul network. Viewed externally, from outside the dashed line in the figure, the local area network appears to be monolithic.

Subnetworking also provides an orderly means for handling growth in traffic. Local area networks perform best, providing high throughput with low delay, when they are not heavily loaded. As traffic on a local area network grows with time, if a higher speed technology is not available, it may be desirable to split the network into two or more interconnected subnetworks. Since the bridges which interconnect the subnetworks are selective in their repeating of packets "across the bridge," not all packets from a subnetwork will flow to all other subnetworks, and the traffic density on each subnetwork will be less than that of the original monolithic network. If the partitioning of the hosts into subnetworks can be done along the lines of "communities of interest," such that a group of hosts with high traffic rates among themselves but with substantially lower traffic rates to other hosts are placed in the same subnetwork, traffic across the bridges will be minimized, and a greater fraction of all packets will stay within their subnetwork of origin.

C. Bridges

A bridge, depicted in Fig. 11, contains:

two network interfaces, one appropriate to each of the two subnetworks it interconnects,
a limited amount of packet buffer memory, and
a control element, which implements an appropriate filter function to decide which messages to "pull off" one subnetwork and buffer until it has an opportunity to retransmit it to the other subnetwork.

The topology of the subnetworks interconnected by a bridge determines the complexity of its filter function. A bridge with a simple filter function can be implemented using a finite state machine as its control element; a complex filter function, which may involve a periodic exchange of information among bridges on the network to determine correct routing, may require the capabilities of a microprocessor [34].

A bridge *must* buffer packets since, upon receiving a message from one subnetwork which it decides to repeat to the other

[5] The subnetwork concept, as we describe it, is a generalization of an approach suggested by Pierce [5] for use with multiple loops or rings.

[6] The concept of the filter function is introduced in the "filtering repeaters" described by Boggs and Metcalfe [14].

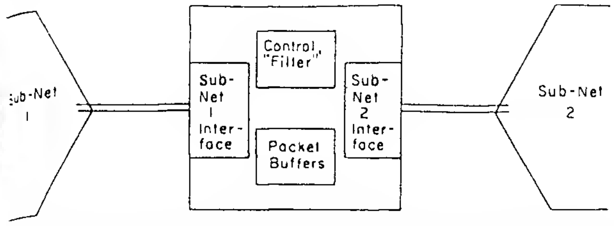


Fig. 11. The structure of a bridge. A bridge would most naturally be located at a point where the two subnetworks it interconnects have been made physically adjacent.

subnetwork, it must wait for an opportunity to transmit on that subnetwork, according to the control structure of that subnetwork. Packet buffers also aid a bridge in handling instantaneous cross-bridge traffic peaks during which the traffic offered by one subnetwork exceeds the available capacity of the other. This situation can arise if the bridge interconnects subnetworks of dissimilar data transmission rates, or subnetworks of drastically different traffic densities. However, if the sustained cross-bridge traffic offered is greater than the target subnetwork can handle, the bridge must discard packets. This is an acceptable course of action, as local area network protocols are generally prepared to handle lost packets.

D. *Transparency*

The subnetwork structure of a local area network should be transparent, both to the hosts on the local area network and to the "outside world"—other networks to which the local area network may be connected via gateways. A host on the local area network wishing to transmit a packet to another should have no knowledge of whether that host is on the same subnetwork, in which case the packet will be received by the destination host directly, or whether the destination host is on another subnetwork, in which case the packet is retransmitted by one or more bridges. In particular, no ordinary data packets *are ever addressed* to a bridge; rather, packets are simply addressed to their destination hosts, and may be picked up by a bridge and passed along through other subnetworks, finally reaching their destinations. This is a key distinction between subnetworking, with bridges, and internetworking, with gateways: in internetworking, a host about to transmit a packet must realize that the host to which it is addressed is on a different network. The sending host must transmit the message in a local network "wrapper" to an appropriate gateway, which "unwraps" it, performs protocol conversions, if any, packet fragmentation, etc., as necessary, and then transmits the message into the other network. In subnetworking, protocols are identical over all subnetworks, and packet sizes are compatible, so that neither protocol conversion nor fragmentation takes place in the bridges. Finally, as was mentioned above, a packet is directly addressed to its destination host, not to a bridge, for hosts do not know that the local area network is composed of subnetworks.

E. *Impact on Network Characteristics*

Splitting a local area network into subnetworks has little impact on the key characteristics of the network. From the point of view of the users and hosts of the network, addressing is affected only slightly, if at all. Bridges must determine whether or not a packet should be picked up for retransmission; one way to aid bridges in this determination is to include a subnetwork field in the address of each host. Other routing techniques which have no impact at all on addressing (such as complete table look-up of host addresses by the bridges) can be implemented, although usually at the expense of greater complexity within the bridges.

Splitting a local area network into subnetworks should have no effect on the protocols of the network. One exception is if a particular subnetwork technology provides a hardware acknowledgment of delivery of a packet (as in the DCS Ring Network) [2]; this acknowledgment may only indicate successful receipt by a bridge. However, not all network technologies provide hardware acknowledgments, and, in a network of mixed technologies, host-to-host acknowledgments will generally be provided by software protocols. Traffic is, of course, affected by subnetworking in a positive way. Splitting a local area network into subnetworks in a judicious way can minimize the overall traffic of the network; bottlenecks can be eliminated by using higher bandwidth technologies for affected subnetworks.

F. *The Long-Distance Bridge*

There are situations in which it is necessary to interconnect two subnetworks of a local area network which cannot be brought physically adjacent to one another so that an ordinary bridge may be connected between them. An example of this would be a local area network on a university campus, with a major research laboratory across town. The laboratory may be beyond the range of a twisted-pair ring network or a coaxial cable contention bus network; or it may be within range, but it may be impossible for the university to install its own cables between them.[7] The off campus research laboratory can be given its own subnetwork, connected to the main campus subnetwork via a specialized *long-distance bridge*.

A long-distance bridge is made up of two *half-bridges* at either end of a suitable full-duplex point-to-point communication link, such as a high-bandwidth common carrier circuit, an optical link, or a private microwave link (Fig. 12). Some other network technology such as packet radio could be used to derive this point-to-point link as desired.[8] Each half-bridge contains an appropriate interface to its subnetwork, packet buffers, and a controller. In addition to its filtering function, the controller of a half-bridge regulates the flow of data over the communication link between the two halves of the bridge. Of course, it is possible that the bridge communication link may be of lower bandwidth than the two subnetworks it interconnects. Additional packet buffers at each half-bridge can help to smoothe out traffic peaks, but if the communication link is a bottleneck, the long-distance bridge must discard packets just as an ordinary bridge does when it is overloaded.[9]

[7] Although common carriers such as the Bell System operating companies are moving in the direction of leasing wire pairs for transmission of digital signals with customer-provided equipment, these circuits are not intended for use at the high bandwidth of local area networks, and are generally routed through central offices rather than point-to-point.

[8] Although we do not discuss it further in this paper, there is an interesting philosophical issue whether the intervening network should be viewed in the internetworking context using gateways or as a point-to-point link within a single bridge.

[9] If the bottleneck created by the communication link of a long bridge is severe, the local area network advantages of high-bandwidth communication with low delay will be forfeited.

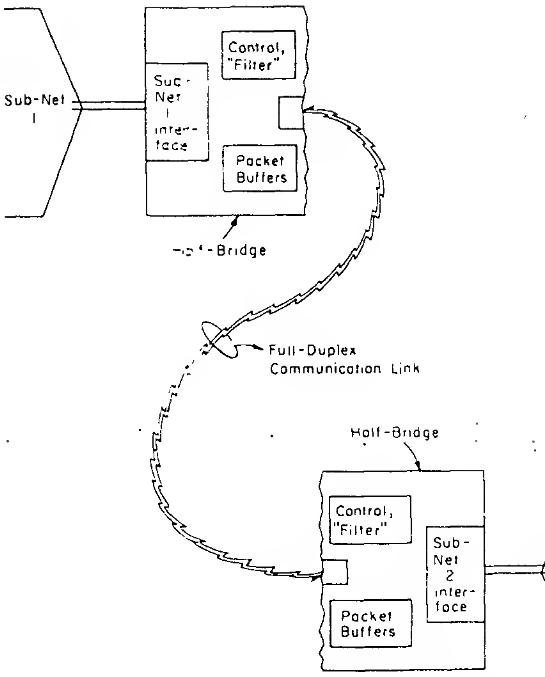


Fig. 12. The "long bridge." In this case, the two subnetworks cannot be made physically adjacent, so a half-bridge is attached to each, and a full-duplex communication link is employed to interconnect the two half-bridges. The control and filter functions, and the packet buffers, are replicated in each half-bridge.

VIII. Conclusion

The utilization of a technological innovation often occurs in two stages. In the first stage, the innovation is exploited to perform better the same tasks that were already being performed. In the second stage, new applications are discovered, which could not be reasonably performed or even forseen prior to the innovation. Local area networks are now on the threshold of this second stage. While there is still much room for creativity in improving the innovation itself—reducing the cost of the network interface and increasing its speed and convenience—the real challenge lies in identifying new sorts of applications that a local area network can make possible.

Current trends in hardware costs encourage abandonment of a single large computer in favor of a number of smaller machines. This decentralization of computing power is, for many applications, a natural and obvious pattern. In many information processing applications, for example, the information itself is distributed in nature, and can most appropriately be managed by distributed machines. Distributed applications can only be constructed, however, if it is possible to link their machines together in an effective manner. Subject to their geographical limitations, local area networks offer a very effective and inexpensive way to provide this interconnection. The greatest impact of local area networks will come with the development of operating systems that integrate the idea of distribution and communication at a fundamental level.

The impact of local area networks on the decentralization of computing is sociological as well as technological. Operational control of centralized computers has traditionally been vested in the staff of a computer center. The trend toward decentralized computing greatly increases the autonomy of individual managers in the operation of their machines, and appears to reduce the need for a centralized staff of computer managers. The communication capability made available by local area networks will serve to bind these decentralized machines together into a unified information processing resource. The effectiveness of this resource can be measured by the degree of coherence it achieves, which, in turn, depends upon the care and foresight put into the design of the local area network and the development of standards for communication at all levels. It is in the identification of areas in which standards are needed, and in their development, that the staff of the "computer center" of the future will find its work.

References

[1] D. J. Farber and K. C. Larson, "The system architecture of the distributed computer system—The communications system," presented at the *Symposium on Computer Networks* (Polytechnic Institute of Brooklyn, Brooklyn, NY, Apr. 1972)

[2] A. G. Fraser, "On the interface between computers and data communications systems," *Commun. Ass. Comput. Mach.*, pp. 31–34, July 15, 1969.

[3] IEEE Instrumentation and Measurements Group *IEEE Standard Digital Interface for Programmable Instrumentation*, IEEE Standard 488, 1975.

[4] D. C. Loomis, "Ring communication protocols," University of California, Department of Information and Computer Science, Irvine, CA, Tech. Rep. 26, Jan. 1973.

[5] J. R. Pierce, "Network for block switching of data," *Bell Syst. Tech. J.*, vol. 51, pp. 1133–1143, July/Aug. 1972.

[6] A. Hopper, "Data ring at computer laboratory, University of Cambridge," in *Computer Science and Technology: Local Area Networking*. Washington, DC, Nat. Bur. Stand., NBS Special Publ. 500-31, Aug. 22–23, 1977, pp. 11–16.

[7] P. Zafiropulo and E. H. Rothauser, "Signalling and frame structures in highly decentralized loop systems," in *Proc. Int. Conf. on Computer Communication* (Washington, DC), IBM Res. Lab., Zurich, Switzerland, pp. 309–315.

[8] G. Babic and T. L. Ming, "A performance study of the distributed loop computer network (DCLN)," in *Proc. Computer Networking Symp.*, (National Bureau of Standards, Gaithersburg, MD, December 15, 1977), pp. 66–76.

[9] E. R. Hafner *et al.*, "A digital loop communication system," *IEEE Trans. Commun.*, p. 877, June 1974.

[10] M. V. Wilkes, "Communication using a digital ring," in *Proc. PACNET Conf.* (Sendai, Japan, August 1975), pp. 47–56.

[11] M. T. Liu and C. C. Reames, "Message communication protocol and operating system design for the distributed loop computer network (DLCN)," in *Proc. 4th Annu. Symp. Computer Architecture*, pp. 193–200, Mar. 1977.

[12] N. Abramson, "The ALOHA system," University of Hawaii Tech. Rep. No. B72-1, Jan. 1972; also *Computer Communication Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[13] R. M. Metcalfe, "Packet communication," M.I.T., Project MAC Tech. Rep. 114, Cambridge, MA, Dec. 1973.

[14] D. R. Boggs and R. M. Metcalfe, "Ethernet: Distributed packet switching for local computer networks," *Comm. Ass. Comput. Mach.*, vol. 19, no. 7, pp. 395–404, July 1976.

[15] E. D. Jensen, "The Honeywell experimental distributed processor—An overview," *Computer*, Jan. 1978.

[16] Digital Equipment Corporation, *PDP-11 Processor Handbook*. Maynard, MA: Digital Equipment Corporation, 1973.

[17] W. D. Farmer and E. E. Newhall, "An experimental distributed switching system to handle bursty computer traffic," in *Proc. ACM Symp. Problems in the Optimization of Data Communication Systems* (Pine Mountain, GA, Oct. 1969), pp. 31–34.

[18] A. G. Fraser, "A virtual channel network," *Datamation*, pp. 51–56, Feb. 1975.

[19] *IBM Synchronous Data Link Control General Information*, GA27-3093-0, File GENL-09, IBM Systems Development Division, Publications Center, North Carolina, 1974.

[20] P. Mockapetris and D. J. Farber, "Experiences with the distributed computer system," submitted to the *J. Distributed Processing*, 1978.

[21] P. Mockapetris, "Design considerations and implementation of the ARPA LNI name table," Univ. California, Dep. Information and Computer Sci., Tech. Rep. 92, Irvine, CA, Apr. 1978.

[22] D. G. Willard, "A time division multiple access system for digital communication," *Comput. Des.*, vol. 13, no. 6, pp. 79–83, June 1974.

N. B. Meisner *et al.*, "Time division digital bus techniques implemented on coaxial cable," in *Proc. Computer Networking Symp.* (National Bureau of Standards, Gaithersburg, MD, Dec. 15, 1977).

R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kurnzelman, "Advances in packet radio technology," this issue, pp. 1468-1496.

P. Mockapetris *et al.*, "On the design of local network interfaces," *Informat. Process.*, vol. 77, pp. 427-430, Aug. 1977.

ARPANET Protocol Handbook, Network Information Center, SRI International, Menlo Park, CA, NIC 7014, revised Jan. 1978.

V. Cerf and R. Kalin, "A protocol for packet network interconnector," *IEEE Trans. Commun.*, vol. COM-25, No. 1, pp. 169-178, May 1974.

L. Pouzin, "Virtual circuits vs. datagrams–Technical and political problems," in *AFIPS Conf. Proc.* (National Computer Conf., June 1976), p. 483.

D. H. Crocker *et al.*, "Standard for the format of ARPA network text messages," ARPA Network RFC 733, NIC 41952, Nov. 21, 1977.

[30] R. H. Thomas, "A resource sharing executive for the ARPANET," *AFIPS Conf. Proc.*, vol. 42 (Nat. Computer Conf. and Exposition, 1973), pp. 155-163.

[31] D. J. Farber and F. H. Heinrich, "The structure of a distributed computer system–The distributed file system," in *Proc. Int. Conf. on Computer Communication* (Washington, DC, 1972), pp. 364-370.

[32] E. G. Manning and R. W. Peebles, "A homogeneous network for data sharing communications," Computer Communications Network Group, University of Waterloo, Waterloo, ON, Tech. Rep. CCNG-E-12, Mar. 1974.

[33] V. G. Cerf and P. T. Kirstein, "Issues in packet network interconnection," this issue, pp. 1386-1408.

[34] S. L. Ratliff, "A dynamic routing algorithm for a local packet network," S.B. thesis, M.I.T., Department of Electrical Engineering and Computer Science, Cambridge, MA, Feb. 1978.

Enhanced Message Addressing Capabilities for Computer Networks

JOHN M. McQUILLAN, MEMBER, IEEE

Invited Paper

Abstract–Three message addressing modes are described:

Logical addressing, in which a permanently assigned address denotes one or more physical addresses. This permits multiple connections of the subscriber to the network, as well as other functions.

Broadcast addressing, in which a message is addressed to all subscribers.

Group addressing and multidestination addressing, in which a message carries the name of a list of addresses, or the list itself.

These methods facilitate many new ways of using computer networks. The paper focuses on two basic issues for each method: efficiency and reliability, and recommends implementation approaches in each case. Significant performance improvements are possible if these addressing methods are implemented with efficient delivery mechanisms. A distinction is made between virtual circuit and datagram systems; virtual circuits are superior for logical addressing, while datagrams are preferable for broadcast, group, and multidestination addressing.

I. Introduction

HOW SHOULD one user of a network address messages to other users? The answer to this question is fundamental in defining the appearance of the network to its users. For example, does one user have to know exactly where the other is located, or just the region of the network, or is the address independent of location? Can he identify himself to the network or does the network know who he is automatically? If self-identification is possible, can he have several addresses corresponding to several roles or functions? Can he have multiple connections to the network, and can he move from one location to another without changing his address(es)? Can he send a single message to a group or list of other users (e.g., a mailing list) automatically? Can he set up "conference calls" with other users, and join conferences in progress? Can he send a message to all other users?

These questions are important for several reasons: some addressing modes allow functions which would not be available otherwise (e.g., the ability to send a message to a distribution list without knowing the identity or location of the members of the list), and which are essential for certain types of users and applications. Furthermore, these addressing capabilities offer opportunities for efficient implementations that would not exist otherwise (e.g., a message addressed to a group can be transmitted with fewer packets than the equivalent separately addressed messages). The topic of addressing has received surprisingly little attention to date; the present paper indicates that it may be a fruitful area for further work.

Manuscript received May 15, 1978; revised July 3, 1978.
The author is with Bolt Beranek and Newman, Inc., Cambridge, MA 02138.

N. B. Meisner *et al.*, "Time division digital bus techniques implemented on coaxial cable," in *Proc. Computer Networking Symp.* (National Bureau of Standards, Gaithersburg, MD, Dec. 15, 1977).

R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kurzelman, "Advances in packet radio technology," this issue, pp. 1468-1496.

P. Mockapetris *et al.*, "On the design of local network interfaces," *Informat. Process.*, vol. 77, pp. 427-430, Aug. 1977.

ARPANET Protocol Handbook, Network Information Center, SRI International, Menlo Park, CA, NIC 7014, revised Jan. 1978.

V. Cerf and R. Kalin, "A protocol for packet network interconnector," *IEEE Trans. Commun.*, vol. COM-25, No. 1, pp. 169-178, May 1974.

L. Pouzin, "Virtual circuits vs. datagrams—Technical and political problems," in *AFIPS Conf. Proc.* (National Computer Conf., June 1976), p. 483.

D. H. Crocker *et al.*, "Standard for the format of ARPA network text messages," ARPA Network RFC 733, NIC 41952, Nov. 21, 1977.

[30] R. H. Thomas, "A resource sharing executive for the ARPANET," *AFIPS Conf. Proc.*, vol. 42 (Nat. Computer Conf. and Exposition, 1973), pp. 155-163.

[31] D. J. Farber and F. H. Heinrich, "The structure of a distributed computer system—The distributed file system," in *Proc. Int. Conf. on Computer Communication* (Washington, DC, 1972), pp. 364-370.

[32] E. G. Manning and R. W. Peebles, "A homogeneous network for data sharing communications," Computer Communications Network Group, University of Waterloo, Waterloo, ON, Tech. Rep. CCNG-E-12, Mar. 1974.

[33] V. G. Cerf and P. T. Kirstein, "Issues in packet network interconnection," this issue, pp. 1386-1408.

[34] S. L. Ratliff, "A dynamic routing algorithm for a local packet network," S.B. thesis, M.I.T., Department of Electrical Engineering and Computer Science, Cambridge, MA, Feb. 1978.

Enhanced Message Addressing Capabilities for Computer Networks

JOHN M. McQUILLAN, MEMBER, IEEE

Invited Paper

stract—Three message addressing modes are described:

Logical addressing, in which a permanently assigned address de- es one or more physical addresses. This permits multiple connections the subscriber to the network, as well as other functions.

Broadcast addressing, in which a message is addressed to all sub- ers.

Group addressing and multidestination addressing, in which a mes- carries the name of a list of addresses, or the list itself.

ese methods facilitate many new ways of using computer networks. paper focuses on two basic issues for each method: efficiency and ility, and recommends implementation approaches in each case. ficant performance improvements are possible if these addressing ods are implemented with efficient delivery mechanisms. A dis- on is made between virtual circuit and datagram systems; virtual its are superior for logical addressing, while datagrams are prefer- for broadcast, group, and multidestination addressing.

I. Introduction

HOW SHOULD one user of a network address messages to other users? The answer to this question is fundamental in defining the appearance of the network to its s. For example, does one user have to know exactly where the other is located, or just the region of the network, or is the address independent of location? Can he identify himself to the network or does the network know who he is automatically? If self-identification is possible, can he have several addresses corresponding to several roles or functions? Can he have multiple connections to the network, and can he move from one location to another without changing his address(es)? Can he send a single message to a group or list of other users (e.g., a mailing list) automatically? Can he set up "conference calls" with other users, and join conferences in progress? Can he send a message to all other users?

These questions are important for several reasons: some addressing modes allow functions which would not be available otherwise (e.g., the ability to send a message to a distribution list without knowing the identity or location of the members of the list), and which are essential for certain types of users and applications. Furthermore, these addressing capabilities offer opportunities for efficient implementations that would not exist otherwise (e.g., a message addressed to a group can be transmitted with fewer packets than the equivalent separately addressed messages). The topic of addressing has received surprisingly little attention to date; the present paper indicates that it may be a fruitful area for further work.

nuscript received May 15, 1978; revised July 3, 1978.

author is with Bolt Beranek and Newman, Inc., Cambridge, MA 38.

0018-9219/78/1100-1517$00.75 © 1978 IEEE

tic Co
Comm
ate C
Signa

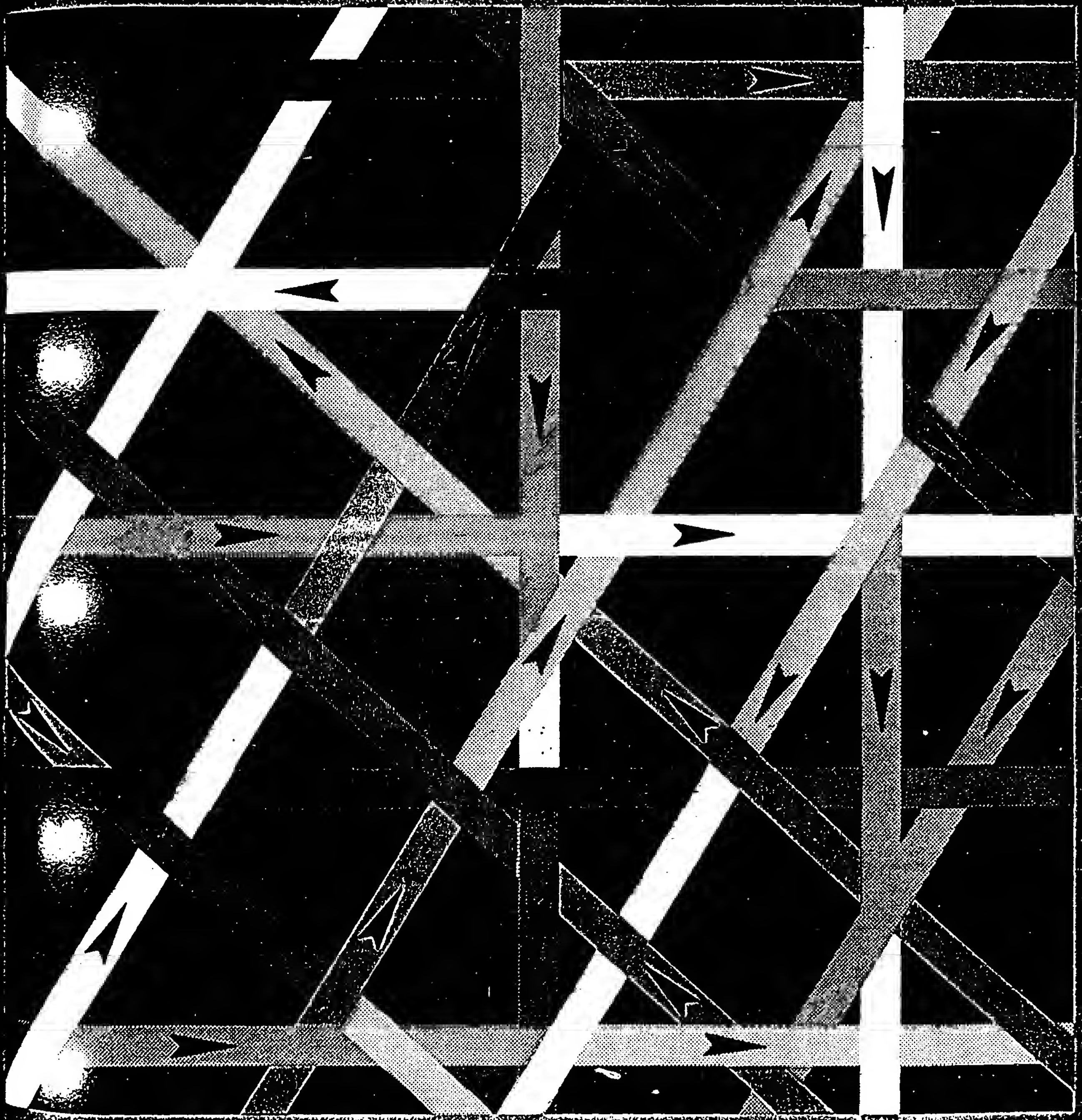
y Ph
er S

PROCEEDINGS OF THE IEEE

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS

NOVEMBER 1978

SPECIAL ISSUE ON packet communication networks

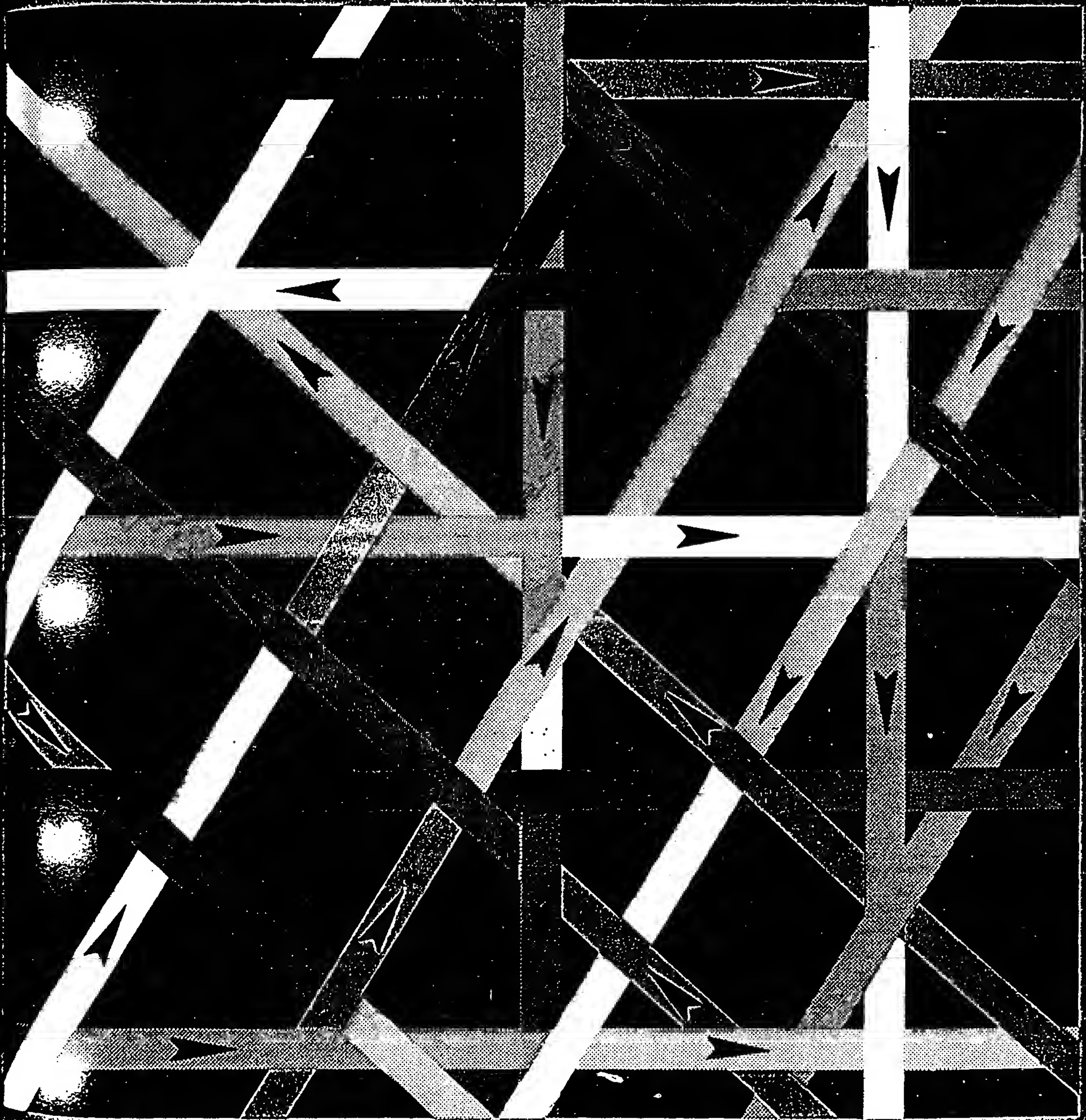


PROCEEDINGS OF THE IEEE

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS

NOVEMBER 1978

SPECIAL ISSUE ON packet communication networks



ific Co
Comm
ate D
Signa

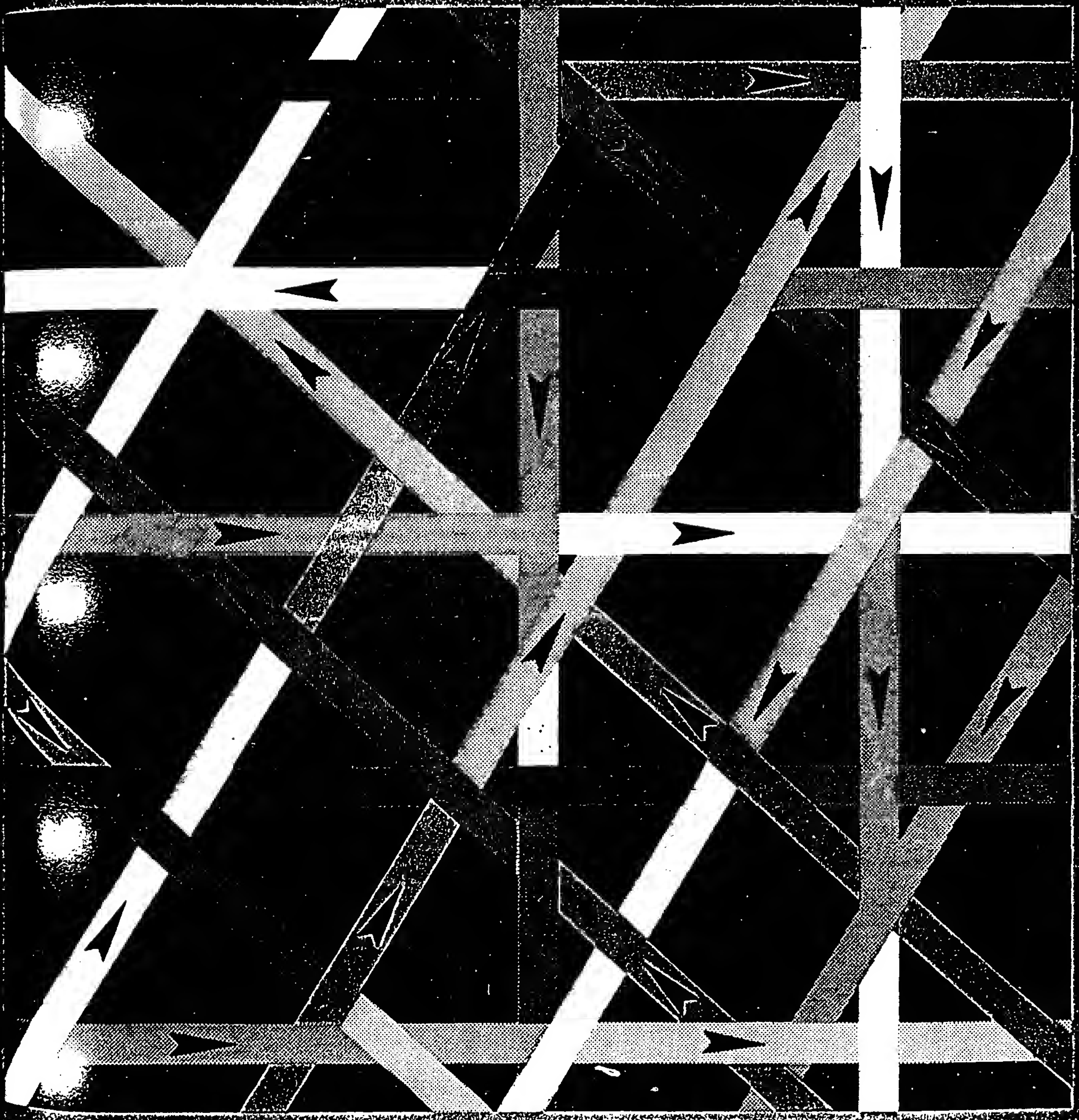
y Ph
er S

PROCEEDINGS OF THE IEEE

THE INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS

NOVEMBER 1978

SPECIAL ISSUE ON packet communication networks



N. B. Meisner *et al.*, "Time division digital bus techniques implemented on coaxial cable," in *Proc. Computer Networking Symp.* (National Bureau of Standards, Gaithersburg, MD, Dec. 15, 1977).

R. E. Kahn, S. A. Gronemeyer, J. Burchfiel, and R. C. Kurzelman, "Advances in packet radio technology," this issue, pp. 1468-1496.

P. Mockapetris *et al.*, "On the design of local network interfaces," *Informat. Process.*, vol. 77, pp. 427-430, Aug. 1977.

ARPANET Protocol Handbook, Network Information Center, SRI International, Menlo Park, CA, NIC 7014, revised Jan. 1978.

V. Cerf and R. Kalin, "A protocol for packet network interconnector," *IEEE Trans. Commun.*, vol. COM-25, No. 1, pp. 169-178, May 1974.

L. Pouzin, "Virtual circuits vs. datagrams–Technical and political problems," in *AFIPS Conf. Proc.* (National Computer Conf., June 1976), p. 483.

D. H. Crocker *et al.*, "Standard for the format of ARPA network text messages," ARPA Network RFC 733, NIC 41952, Nov. 21, 1977.

[30] R. H. Thomas, "A resource sharing executive for the ARPANET," *AFIPS Conf. Proc.*, vol. 42 (Nat. Computer Conf. and Exposition, 1973), pp. 155-163.

[31] D. J. Farber and F. H. Heinrich, "The structure of a distributed computer system–The distributed file system," in *Proc. Int. Conf. on Computer Communication* (Washington, DC, 1972), pp. 364-370.

[32] E. G. Manning and R. W. Peebles, "A homogeneous network for data sharing communications," Computer Communications Network Group, University of Waterloo, Waterloo, ON, Tech. Rep. CCNG-E-12, Mar. 1974.

[33] V. G. Cerf and P. T. Kirstein, "Issues in packet network interconnection," this issue, pp. 1386-1408.

[34] S. L. Ratliff, "A dynamic routing algorithm for a local packet network," S.B. thesis, M.I.T., Department of Electrical Engineering and Computer Science, Cambridge, MA, Feb. 1978.

Enhanced Message Addressing Capabilities for Computer Networks

JOHN M. McQUILLAN, MEMBER, IEEE

Invited Paper

stract–Three message addressing modes are described:

Logical addressing, in which a permanently assigned address de-s one or more physical addresses. This permits multiple connections the subscriber to the network, as well as other functions.

Broadcast addressing, in which a message is addressed to all sub-ers.

Group addressing and multidestination addressing, in which a mes- carries the name of a list of addresses, or the list itself.

se methods facilitate many new ways of using computer networks. paper focuses on two basic issues for each method: efficiency and ility, and recommends implementation approaches in each case. ficant performance improvements are possible if these addressing ods are implemented with efficient delivery mechanisms. A dis-on is made between virtual circuit and datagram systems; virtual its are superior for logical addressing, while datagrams are prefer- for broadcast, group, and multidestination addressing.

I. Introduction

HOW SHOULD one user of a network address messages to other users? The answer to this question is fundamental in defining the appearance of the network to its . For example, does one user have to know exactly where the other is located, or just the region of the network, or is the address independent of location? Can he identify himself to the network or does the network know who he is automatically? If self-identification is possible, can he have several addresses corresponding to several roles or functions? Can he have multiple connections to the network, and can he move from one location to another without changing his address(es)? Can he send a single message to a group or list of other users (e.g., a mailing list) automatically? Can he set up "conference calls" with other users, and join conferences in progress? Can he send a message to all other users?

These questions are important for several reasons: some addressing modes allow functions which would not be available otherwise (e.g., the ability to send a message to a distribution list without knowing the identity or location of the members of the list), and which are essential for certain types of users and applications. Furthermore, these addressing capabilities offer opportunities for efficient implementations that would not exist otherwise (e.g., a message addressed to a group can be transmitted with fewer packets than the equivalent separately addressed messages). The topic of addressing has received surprisingly little attention to date; the present paper indicates that it may be a fruitful area for further work.

uscript received May 15, 1978; revised July 3, 1978.
author is with Bolt Beranek and Newman, Inc., Cambridge, MA 38.

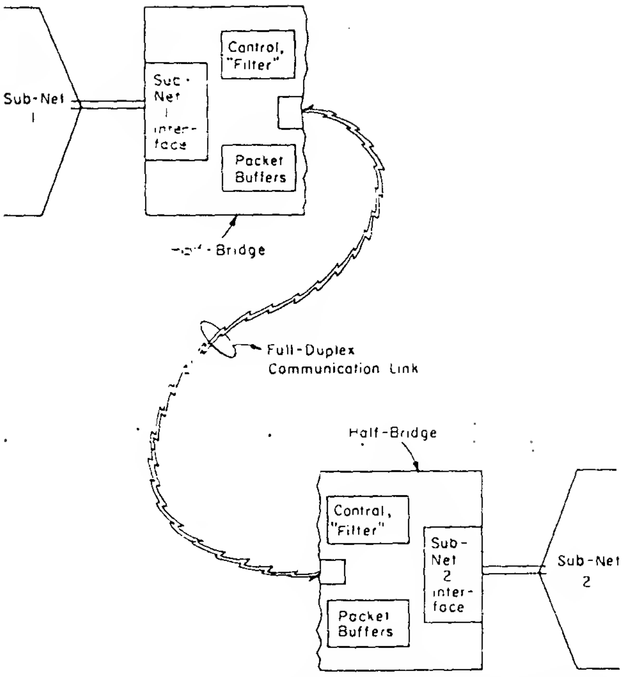


Fig. 12. The "long bridge." In this case, the two subnetworks cannot be made physically adjacent, so a half-bridge is attached to each, and a full-duplex communication link is employed to interconnect the two half-bridges. The control and filter functions, and the packet buffers, are replicated in each half-bridge.

VIII. Conclusion

The utilization of a technological innovation often occurs in two stages. In the first stage, the innovation is exploited to perform better the same tasks that were already being performed. In the second stage, new applications are discovered, which could not be reasonably performed or even forseen prior to the innovation. Local area networks are now on the threshold of this second stage. While there is still much room for creativity in improving the innovation itself—reducing the cost of the network interface and increasing its speed and convenience—the real challenge lies in identifying new sorts of applications that a local area network can make possible.

Current trends in hardware costs encourage abandonment of a single large computer in favor of a number of smaller machines. This decentralization of computing power is, for many applications, a natural and obvious pattern. In many information processing applications, for example, the information itself is distributed in nature, and can most appropriately be managed by distributed machines. Distributed applications can only be constructed, however, if it is possible to link their machines together in an effective manner. Subject to their geographical limitations, local area networks offer a very effective and inexpensive way to provide this interconnection. The greatest impact of local area networks will come with the development of operating systems that integrate the idea of distribution and communication at a fundamental level.

The impact of local area networks on the decentralization of computing is sociological as well as technological. Operational control of centralized computers has traditionally been vested in the staff of a computer center. The trend toward decentralized computing greatly increases the autonomy of individual managers in the operation of their machines, and appears to reduce the need for a centralized staff of computer managers. The communication capability made available by local area networks will serve to bind these decentralized machines together into a unified information processing resource. The effectiveness of this resource can be measured by the degree of coherence it achieves, which, in turn, depends upon the care and foresight put into the design of the local area network and the development of standards for communication at all levels. It is in the identification of areas in which standards are needed, and in their development, that the staff of the "computer center" of the future will find its work.

References

[1] D. J. Farber and K. C. Larson, "The system architecture of the distributed computer system—The communications system," presented at the *Symposium on Computer Networks* (Polytechnic Institute of Brooklyn, Brooklyn, NY, Apr. 1972)

[2] A. G. Fraser, "On the interface between computers and data communications systems," *Commun. Ass. Comput. Mach.*, pp. 31–34, July 15, 1969.

[3] IEEE Instrumentation and Measurements Group, *IEEE Standard Digital Interface for Programmable Instrumentation*, IEEE Standard 488, 1975.

[4] D. C. Loomis, "Ring communication protocols," University of California, Department of Information and Computer Science, Irvine, CA, Tech. Rep. 26, Jan. 1973.

[5] J. R. Pierce, "Network for block switching of data," *Bell Syst. Tech. J.*, vol. 51, pp. 1133–1143, July/Aug. 1972.

[6] A. Hopper, "Data ring at computer laboratory, University of Cambridge," in *Computer Science and Technology: Local Area Networking*. Washington, DC, Nat. Bur. Stand., NBS Special Publ. 500-31, Aug. 22–23, 1977, pp. 11–16.

[7] P. Zafiropulo and E. H. Rothauser, "Signalling and frame structures in highly decentralized loop systems," in *Proc. Int. Conf. on Computer Communication* (Washington, DC), IBM Res. Lab., Zurich, Switzerland, pp. 309–315.

[8] G. Babic and T. L. Ming, "A performance study of the distributed loop computer network (DCLN)," in *Proc. Computer Networking Symp.*, (National Bureau of Standards, Gaithersburg, MD, December 15, 1977), pp. 66–76.

[9] E. R. Hafner *et al.*, "A digital loop communication system," *IEEE Trans. Commun.*, p. 877, June 1974.

[10] M. V. Wilkes, "Communication using a digital ring," in *Proc. PACNET Conf.* (Sendai, Japan, August 1975), pp. 47–55.

[11] M. T. Liu and C. C. Reames, "Message communication protocol and operating system design for the distributed loop computer network (DLCN)," in *Proc. 4th Annu. Symp. Computer Architecture*, pp. 193–200, Mar. 1977.

[12] N. Abramson, "The ALOHA system," University of Hawaii Tech. Rep. No. B72-1, Jan. 1972; also *Computer Communication Networks*. Englewood Cliffs, NJ: Prentice-Hall, 1973.

[13] R. M. Metcalfe, "Packet communication," M.I.T., Project MAC Tech. Rep. 114, Cambridge, MA, Dec. 1973.

[14] D. R. Boggs and R. M. Metcalfe, "Ethernet: Distributed packet switching for local computer networks," *Comm. Ass. Comput. Mach.*, vol. 19, no. 7, pp. 395–404, July 1976.

[15] E. D. Jensen, "The Honeywell experimental distributed processor—An overview," *Computer*, Jan. 1978.

[16] Digital Equipment Corporation, *PDP-11 Processor Handbook*. Maynard, MA: Digital Equipment Corporation, 1973.

[17] W. D. Farmer and E. E. Newhall, "An experimental distributed switching system to handle bursty computer traffic," in *Proc. ACM Symp. Problems in the Optimization of Data Communication Systems* (Pine Mountain, GA, Oct. 1969), pp. 31–34.

[18] A. G. Fraser, "A virtual channel network," *Datamation*, pp. 51–56, Feb. 1975.

[19] *IBM Synchronous Data Link Control General Information*, GA27-3093-0, File GENL-09, IBM Systems Development Division, Publications Center, North Carolina, 1974.

[20] P. Mockapetris and D. J. Farber, "Experiences with the distributed computer system," submitted to the *J. Distributed Processing*, 1978.

[21] P. Mockapetris, "Design considerations and implementation of the ARPA LNI name table," Univ. California, Dep. Information and Computer Sci., Tech. Rep. 92, Irvine, CA, Apr. 1978.

[22] D. G. Willard, "A time division multiple access system for digital communication," *Comput. Des.*, vol. 13, no. 6, pp. [illegible] June 1974.

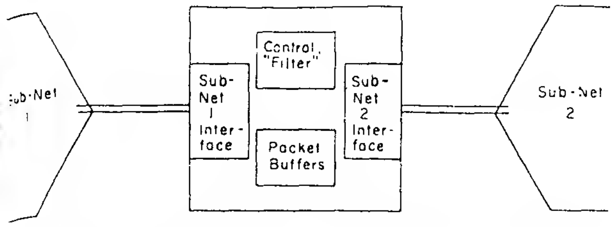


Fig. 11. The structure of a bridge. A bridge would most naturally be located at a point where the two subnetworks it interconnects have been made physically adjacent.

subnetwork, it must wait for an opportunity to transmit on that subnetwork, according to the control structure of that subnetwork. Packet buffers also aid a bridge in handling instantaneous cross-bridge traffic peaks during which the traffic offered by one subnetwork exceeds the available capacity of the other. This situation can arise if the bridge interconnects subnetworks of dissimilar data transmission rates, or subnetworks of drastically different traffic densities. However, if the sustained cross-bridge traffic offered is greater than the target subnetwork can handle, the bridge must discard packets. This is an acceptable course of action, as local area network protocols are generally prepared to handle lost packets.

D. Transparency

The subnetwork structure of a local area network should be transparent, both to the hosts on the local area network and to the "outside world"—other networks to which the local area network may be connected via gateways. A host on the local area network wishing to transmit a packet to another should have no knowledge of whether that host is on the same subnetwork, in which case the packet will be received by the destination host directly, or whether the destination host is on another subnetwork, in which case the packet is retransmitted by one or more bridges. In particular, no ordinary data packets *are ever addressed* to a bridge; rather, packets are simply addressed to their destination hosts, and may be picked up by a bridge and passed along through other subnetworks, finally reaching their destinations. This is a key distinction between subnetworking, with bridges, and internetworking, with gateways: in internetworking, a host about to transmit a packet must realize that the host to which it is addressed is on a different network. The sending host must transmit the message in a local network "wrapper" to an appropriate gateway, which "unwraps" it, performs protocol conversions, if any, packet fragmentation, etc., as necessary, and then transmits the message into the other network. In subnetworking, protocols are identical over all subnetworks, and packet sizes are compatible, so that neither protocol conversion nor fragmentation takes place in the bridges. Finally, as was mentioned above, a packet is directly addressed to its destination host, not to a bridge, for hosts do not know that the local area network is composed of subnetworks.

E. Impact on Network Characteristics

Splitting a local area network into subnetworks has little impact on the key characteristics of the network. From the point of view of the users and hosts of the network, addressing is affected only slightly, if at all. Bridges must determine whether or not a packet should be picked up for retransmission; one way to aid bridges in this determination is to include a subnetwork field in the address of each host. Other routing techniques which have no impact at all on addressing (such as complete table look-up of host addresses by the bridges) can be implemented, although usually at the expense of greater complexity within the bridges.

Splitting a local area network into subnetworks should have no effect on the protocols of the network. One exception is if a particular subnetwork technology provides a hardware acknowledgment of delivery of a packet (as in the DCS Ring Network) [2]; this acknowledgment may only indicate successful receipt by a bridge. However, not all network technologies provide hardware acknowledgments, and, in a network of mixed technologies, host-to-host acknowledgments will generally be provided by software protocols. Traffic is, of course, affected by subnetworking in a positive way. Splitting a local area network into subnetworks in a judicious way can minimize the overall traffic of the network; bottlenecks can be eliminated by using higher bandwidth technologies for affected subnetworks.

F. The Long-Distance Bridge

There are situations in which it is necessary to interconnect two subnetworks of a local area network which cannot be brought physically adjacent to one another so that an ordinary bridge may be connected between them. An example of this would be a local area network on a university campus, with a major research laboratory across town. The laboratory may be beyond the range of a twisted-pair ring network or a coaxial cable contention bus network; or it may be within range, but it may be impossible for the university to install its own cables between them.[7] The off campus research laboratory can be given its own subnetwork, connected to the main campus subnetwork via a specialized *long-distance bridge*.

A long-distance bridge is made up of two *half-bridges* at either end of a suitable full-duplex point-to-point communication link, such as a high-bandwidth common carrier circuit, an optical link, or a private microwave link (Fig. 12). Some other network technology such as packet radio could be used to derive this point-to-point link as desired.[8] Each half-bridge contains an appropriate interface to its subnetwork, packet buffers, and a controller. In addition to its filtering function, the controller of a half-bridge regulates the flow of data over the communication link between the two halves of the bridge. Of course, it is possible that the bridge communication link may be of lower bandwidth than the two subnetworks it interconnects. Additional packet buffers at each half-bridge can help to smoothe out traffic peaks, but if the communication link is a bottleneck, the long-distance bridge must discard packets just as an ordinary bridge does when it is overloaded.[9]

[7] Although common carriers such as the Bell System operating companies are moving in the direction of leasing wire pairs for transmission of digital signals with customer-provided equipment, these circuits are not intended for use at the high bandwidth of local area networks, and are generally routed through central offices rather than point-to-point.

[8] Although we do not discuss it further in this paper, there is an interesting philosophical issue whether the intervening network should be viewed in the internetworking context using gateways or as a point-to-point link within a single bridge.

[9] If the bottleneck created by the communication link of a long bridge is severe, the local area network advantages of high-bandwidth communication with low delay will be forfeited.

the local net. It leaves unanswered the question of how the additional features, such as complex flow control, buffering, and speed matching required for the long-haul protocol should be implemented. One approach would be to implement them in every host that desires to communicate over the long-haul network; this implies a programming burden for every machine. An alternative would be to implement the additional functions in the gateway machine that interconnects the local area network to the long-haul network. This would add considerable complexity to the gateway, for it will have to cope with such problems as the speed differential between the two networks without having the benefit of the flow control mechanisms normally used for this purpose in the long-haul network. At this time, it is not clear whether the gateway can assume the entire responsibility for augmenting a local network virtual circuit protocol with the functions required for communication through a long-haul network.

It would be advantageous to make sure the local area network protocols are also compatible with other communication models, such as single message exchange or selective broadcast, that may become available on commercial long-haul networks in the future. However, this presupposes that the long-haul networks attached to the local area network use a two-layer low-level protocol implementation such as that described for the local area network, and if the long-haul networks do use such an implementation, that they provide an interface that allows direct use of the datagram layer. Many current long-haul networks do not provide that interface.

VII. The Subnetwork Concept

Resting midway between the monolithic, single-technology, local area network and the internetworking environment is an approach to local area networking that we term the *subnetwork concept*, which provides for a mix of network technologies within a uniform addressing and administrative structure.

A. General Approach

A local area network can be composed of a collection of subnetworks, possibly implemented with various network technologies and perhaps with various transmission rates, but using identical software protocols, compatible packet sizes, and a single overall homogeneous address space.[5] These subnetworks are interconnected by *bridges*, which are midway in complexity between the repeaters used in a multisegment contention bus network (ETHERNET) and the gateway processor used between networks in an internetworking environment. This general structure is indicated in Fig. 10. A bridge links two subnetworks, generally at a location at which they are physically adjacent, and selectively repeats packets from each of them to the other, according to a "filter function."[6] In addition, since they buffer the packets they repeat, they can perform a speed-matching function as well.

B. Advantages of Subnetworking

The subnetworking concept enables a variety of technologies and data rates to be utilized in a single local area network, each to its best advantage. For example, a network could be constructed with a contention bus subnetwork, perhaps using coaxial cable originally installed for CATV, and with a ring subnetwork, using twisted pair which can be easily installed in a crowded laboratory environment. These two subnetworks could be of different data rates; the bridge between the two will handle the speed difference between them.

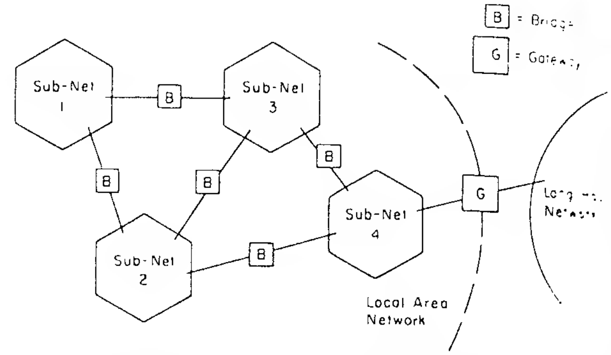


Fig. 10. The subnetwork concept. Here, a local area network is composed of a number of subnetworks, linked in some fashion by bridges. The subnetworks, though of differing technologies, share one address space, and the same protocols are used over the entire network. Thus, the bridges can be simpler than the gateway which connects the local area network to the long-haul network. Viewed externally, from outside the dashed line in the figure, the local area network appears to be monolithic.

Subnetworking also provides an orderly means for handling growth in traffic. Local area networks perform best, providing high throughput with low delay, when they are not heavily loaded. As traffic on a local area network grows with time, if a higher speed technology is not available, it may be desirable to split the network into two or more interconnected subnetworks. Since the bridges which interconnect the subnetworks are selective in their repeating of packets "across the bridge," not all packets from a subnetwork will flow to all other subnetworks, and the traffic density on each subnetwork will be less than that of the original monolithic network. If the partitioning of the hosts into subnetworks can be done along the lines of "communities of interest," such that a group of hosts with high traffic rates among themselves but with substantially lower traffic rates to other hosts are placed in the same subnetwork, traffic across the bridges will be minimized, and a greater fraction of all packets will stay within their subnetwork of origin.

C. Bridges

A bridge, depicted in Fig. 11, contains:

two network interfaces, one appropriate to each of the two subnetworks it interconnects,

a limited amount of packet buffer memory, and

a control element, which implements an appropriate filter function to decide which messages to "pull off" one subnetwork and buffer until it has an opportunity to retransmit it to the other subnetwork.

The topology of the subnetworks interconnected by a bridge determines the complexity of its filter function. A bridge with a simple filter function can be implemented using a finite state machine as its control element; a complex filter function, which may involve a periodic exchange of information among bridges on the network to determine correct routing, can require the capabilities of a microprocessor [34].

A bridge *must* buffer packets since, upon receiving a message from one subnetwork which it decides to repeat to the other

[5] The subnetwork concept, as we describe it, is a generalization of an approach suggested by Pierce [5] for use with multiple loops or rings.

[6] The concept of the filter function is introduced in the "filtering repeaters" described by Boggs and Metcalfe [14].

nade accessible
ice is expensive
having one for
placed remote
; for example,
s.
iputers can be
ices. In fact,
ions from one
computing load
area networks
actical than do
aul networks.
ty of new uses
ng power avail-
r the alternative
arge number of
i small number
xtreme, we can
n each user has
l. Such a com-
by no means
; network, for
on. Data files
tation will need
k to be used as
nmunicate with
is is now done
want access to
ed to each user,
ems, specialized
or connection
features can be

t: The applica-
n be supported
ones already in
logging into a
le Transfer Pro-
[26]. When one
odified to take
al area network,
hat the problem
of modifying the
the network so
ork appear to be
ient of the oper-
the ARPANET,
iser as an explicit
a file from one
ation he may be
achine, and give
l and the foreign
ishes to perform
vo disadvantages
ith moving a file
by the user has
nd the data itself
spent establishing
other site, etc.
outer understands
sible through the
local file system

s in terms of keeping track of the various files that the user cares about, it requires explicit user intervention in order to reach through the network and retrieve a file from another machine. The use of a high-speed local area network will not eliminate any of these problems, but will instead make even more obvious to the user the overhead that the protocol imposes on the transfer of data. Clearly, what is needed is a further integration of the local area network into the file system and user authentication mechanism of the individual operating systems, so that interchange of information between the various machines can he done with less direct user intervention. Some attempts have been made to do this within the context of the ARPANET. RSEXEC is an example of a protocol which makes files on various TENEX operating systems in the ARPANET appear to the user to exist on a single machine [30].

The design of operating system structures to take full advantage of the capabilities of local area networks represents the current edge of research in this area. Examples of operating systems that incorporate a high-speed local area network into their architecture are the Distributed Computing System [31], the Distributed Loop Operating System [11], and MININET [32].

VI. Interconnection of Local Area Networks with other Networks

A. *Motivation for Interconnection*

As was mentioned earlier, a local area network will be only part of the overall communication system used by the hosts attached to it. A very important use of the local area network can be to provide an interconnection between hosts attached to a local area network and other networks such as long-haul packet-switched networks and point-to-point transmission links. The advantage of this method of interconnection is reduced cost, by taking advantage of the fact that connection of a host to a local area network is relatively inexpensive. Instead of connecting all machines directly to the long-haul network, one can connect all the host computers to the local area network, with one machine, the *gateway*, connected to both the local area network and the long-haul network.

B. *Protocol Compatibility*

There are two pitfalls that should be avoided when planning for the interconnection of a local area network with a long-haul network. On the one hand, long-haul networks currently cannot provide all of the functions that local area networks can. If a local area network is initially designed to serve only the function of connecting hosts to a long-haul network, the protocols of the local network may be designed to serve only the needs of communicating with the long-haul network, and may not support the other functions that make a local area network especially attractive. On the other hand, if a local network is initially designed with no thought given to the possibility that it may be interconnected with another network, the protocols designed for it may lack the necessary generality. For example, the addressing structure used on the local area network may not be able to express destinations outside the local network. In either case, the only after-the-fact solution is to implement a second set of protocols for the local area network, so that different protocols are used for intercommunication with long-haul networks and for local services. This proliferation of protocols is undesirable, as it adds to the cost of software development associated with each new host added to the local area network. To avoid these pitfalls, it is important that all the functions a local area network is to provide must be considered from the very inception of the design of the network, and the protocols for the network must be designed to support that entire range of functionality.

Fortunately, initial experiments with protocols for local area networks suggest that a uniform approach to protocol design can support both specialized local network functions and interconnection with other networks, provided that both functions are envisioned from the start. Although the protocols used in the local area network must be made slightly more general to handle the internetworking situation, there is no interference with the realization of the purely local network functions. For example, a more general address field must be used to specify the destination of a message, but the only overhead implied if this same addressing structure is used for purely local messages is additional bits in the message to hold a presumably larger address. Since bandwidth is inexpensive, the bits "wasted" on this larger address are presumably irrelevant.

A slightly more difficult problem, one that is still being studied, is the problem of speed matching between the local area network and the long-haul network. As this paper has characterized the difference between local nets and long-haul nets, it is reasonable to presume that the local network will have a much higher data rate. If a host sends a large number of packets into the local area network with an ultimate destination to be reached through the long-haul network, the packets may arrive at the gateway much faster then the gateway can pass them to the long-haul network. Some mechanism will be required to prevent the gateway from exhausting its buffer space. The speed matching problem is not unique to the gateway between the local area network and the long-haul network; it occurs any time two networks of differing speed are connected together. (The problem may be more extreme here, though, due to the greater speed difference that can be encountered between local area and some long-haul networks. Satellite networks with speeds comparible to local networks are quite conceivable, yet are a long-haul technology.) A general discussion of the problems of internetworking, and some proposed solutions can be found in a companion paper by Cerf and Kirstein in this issue [33].

At the next higher level of protocol, one finds facilities that support various communications models, such as virtual circuits, broadcast, and message exchange. In interconnecting to a long-haul network we are chiefly forced to deal with a virtual circuit model, since that is the only pattern of communication usually supported by commercial long-haul networks. Here, it is appropriate to use a virtual circuit protocol in the local area network as similar as possible to that used in the long-haul network, so that translation between the two is easy. Although there is not as much practical experience available in the area of network interconnection as could be desired, it appears that one can develop a virtual circuit protocol for a local area network that is a compatible subset (in the sense of using compatible packet formats and control algorithms) of a suitable long-haul virtual circuit protocol. This means that it is not necessary to implement two complete virtual circuit protocols, one for internal local network use and the other for communication out through cast message has been successfully received? By one of the possible recipients? By all of the possible recipients? One appropriate strategy is to rely on the high-level application to deal with these problems as a part of its normal operation, rather than have the low-level protocol concern itself with issues of flow control or acknowledgment at all.

3) Protocol Structure: Based on the previous observations, a two-layer structure is a very natural one for low-level protocols in a local area network. The bottom layer should provide the basic function of delivering an addressed message to its (one or many) destinations. This level corresponds to the concept of a *datagram* network [28]. It should also take on the responsibility of detecting that a message has been damaged in transit. To this end it may append a checksum to a message and verify the checksum on receipt. However, this layer probably should not take on the responsibility of ensuring that messages are delivered, and delivered in the order sent, since different applications have different needs and requirements for these functions. The first layer might be implemented entirely in hardware; however, if the packet size, addressing structure, or routing topology of the hardware is not sufficient to provide adequate message size, process addressing, or broadcast selectivity, some software help will be needed to make up the difference.

Above this first layer should be made available a variety of protocols. One protocol should support a virtual circuit mechanism, since a virtual circuit is definitely the appropriate model for a great deal of the communication that will go on in any network, local or otherwise. As alternatives to the virtual circuit protocol, there should be mechanisms for sending isolated messages, for message exchange communication, and additional alternatives to provide support for message models other than the ones we have discussed here. For example, transmission of digitized speech requires a communication model with some but not all of the attributes of the virtual circuit; in particular, reliability is of less concern than timeliness of arrival.

B. Applications of Local Area Networks; Higher Level Protocols

In the previous section we considered low-level protocols for a local area network. These protocols exist, of course, to support higher level protocols, which, in turn, support user applications. In this section we will consider a number of applications for which local area networks are suited.

1) Access to Common Resources: The model of computing most common over the last few years is that of a large centralized computer, with the only remote components being terminals and, perhaps, a few other I/O devices. Line control protocols such as SDLC [19] were created to serve this sort of arrangement. A simple but very important application of a local area network is to generalize this picture very slightly to include more than one central computer. As the total workload grows to exceed the capacity of a single machine, a common solution is to procure a second machine, and to divide the applications and workload between the two. The communication problem to be solved in this arrangement is simple but critical—to allow an individual terminal to have access to both of the central machines. A local area network can solve this problem, and provide some additional capabilities as well. For example, if the central facility has specialized I/O devices such as plotters or microfilm writers, they can be placed on the local area network and made accessible to both central machines—an advantage if a devicc is expensive and is not heavily enough loaded to justify having one for each computer. Further, I/O devices can be placed remote from the central site but convenient to users; for example a line printer can be placed near a cluster of users.

This pattern of sharing among several computers can be expanded to include more than just I/O devices. In fact the network can be used to move computations from one machine to another in order to spread the computing load equally. The high speeds available in the local area networks make this sort of load leveling much more practical than the bandwidths traditionally available on long-haul networks.

2) Decentralized Computing: A wide variety of new uses for a local area network arises if the computing power available is not strongly centralized. Let us consider the alternative of a computing environment consisting of a large number of relatively small machines, each dedicated to a small number of users or a small number of tasks. In the extreme, we can look to the future and imagine the day when each user has a computer on his desk instead of a terminal. Such a completely distributed computing environment by no means eliminates the need for an interconnecting network, for users will still need to exchange information. Data files containing the results of one person's computation will need to be shipped through the local area network to be used as input to other tasks. Users will wish to communicate with each other by exchanging computer mail, as is now done over the ARPANET [29]. Users will still want access to specialized resources which cannot be provided to each user: resources such as large archival storage systems, specialized output devices such as photo typesetters, or connection points to long-haul networks. All of these features can be made available through the local area network.

3) Protocol and Operating System Support: The applications outlined in the previous paragraph can be supported by high-level protocols very similar to the ones already in existence in the ARPANET: TELNET for logging into a remote system through the network, and File Transfer Protocol for exchanging data between machines [26]. When one examines how these protocols might be modified to take advantage of the special attributes of a local area network, for example, its higher speed, one discovers that the problem is not one of modifying the protocols, but of modifying the operating system of the hosts connected to the network so that the services available through the network appear to be a natural part of the programming environment of the operating system. The File Transfer Protocol in the ARPANET, for example, is usually made available to the user as an explicit command which he may invoke to move a file from one machine to another. As part of this invocation he may be required to identify himself at the other machine, and give explicit file names in the syntax of the local and the foreign machine, describing exactly what action he wishes to perform.

This particular view of file transfer has two disadvantages. First, there is a lot of overhead associated with moving a file. Much of the delay in moving the file seen by the user has nothing to do with the time required to send the data itself through the network, but is rather the time spent establishing the connection, identifying the user at the other site, etc. Second, the file system on the local computer understands nothing about the existence of files accessible through the network. No matter how sophisticated the local file system

a) the process to receive the data is scheduled and requests input, or b) the urgent pointer points to data not already received by the process. In case b) an interrupt is sent to the receiving process, indicating that data should be read and processed until the urgent pointer is past. The corresponding mechanism in TCP required that a host be capable of understanding and responding to a special interrupt signal in the data stream, even if the signal had no meaning to the host in its particular application of TCP. The urgent pointer, then, is a simple mechanism that meets the needs of sophisticated host implementations without placing an excessive burden on unsophisticated hosts.

2) *Special Capabilities:* The other aspect of low-level protocols for local area networks to be discussed is the manner in which protocols must be structured to take advantage of, and provide to higher levels, the unique capabilities of local networks. Conventional low-level protocols have provided a function best characterized as a bidirectional stream of bits between two communicating entities—a *virtual circuit.* The virtual circuit is implemented by a process that provides sequenced delivery of packets at the destination. While a virtual circuit is one important form of communication, two others easily provided by a local network are very useful in a variety of contexts. These are *message exchange* communication, where the packets exchanged are not viewed as being members of a sequence of packets but are rather isolated exchanges, and *broadcast* communication in which messages are sent not to one particular recipient but to a selected subset of the potential recipients on the network.

a) Message exchange: A typical example of a message exchange is the situation in which one message asks a question and another provides the answer. For example, if there are a large number of services provided by nodes connected to a local net, it is disadvantageous to maintain, on every node, a table giving all of the addresses of these, for whenever a change is made in the network address of any service, every node's table will need to be revised. Rather, it may be advantageous to maintain, as a network service, a facility which will take the name of a desired entity and give back its network address. Clearly, the pattern of communication with this service is not one of opening a connection and exchanging a large number of messages, but instead is a simple two-message exchange, with a query of the form "What is the address of such and such a service?" and a reply of similarly simple form. While a virtual circuit *could* be used for this exchange, it is unneeded and uses excessive resources.

b) Broadcast: The example given above demonstrates the need for a broadcast mechanism. If the service described above is intended to provide the address of network services, how can we find the address of this service itself? An obvious solution is to broadcast the request for information. The query then takes the form "Would anyone who knows the address of such and such a network service please send it to me!" There are many other examples, some apparently trivial but nonetheless very useful, for support of broadcast queries on a local network. A microprocessor with no calendar clock may broadcast a request for the time of day. A new host attached to the network for the first time may broadcast a message announcing its presence, so that those who maintain tables may discover its existence and record the fact. Broadcast mechanisms in the low-level protocols can also be quite useful in implementing higher level protocols for such applications as document distribution to multiple host nodes, and for speech and video conference calls.

Why are these alternative models of communication not commonly found in traditional networks? The first, and perhaps most important reason is that long-haul networks have not been extensively exploited for applications in which computers directly query other computers with individual, self-contained queries. Instead, the major use of long-haul networks has been for long-term, human-initiated interactions with computers, such as direct terminal use of a remote computer, or long-term attachments of remote job entry stations. Such human interactions usually involve many message exchanges between sender and receiver, so that the extra delay and cost of initial setup of a virtual circuit is insignificant—perhaps even recovered by reducing redundant information in each message. As new applications such as distributed data base systems become more important, these alternative models will become important in long-haul networks, but long-lived connections between terminals and host computers continue to dominate the usage.

The second reason is precisely that discussed in the previous section concerning the relative simplicity of protocols for local area networks—a variety of functions performed in conventional networks are very difficult to understand except in the context of a sequence of ordered messages (a virtual circuit) exchanged between two nodes. For example, flow control is normally handled in network protocols by placing an upper bound on the number of messages which may be flowing at any one time between the sender and the receiver. This concept has meaning only in the restricted case where the sender and the receiver are a well-identified pair exchanging a sequence of messages. There is no obvious equivalent of flow control that can be applied to situations where sender and receiver communicate by sending arbitrary unsequenced messages, or where a sender broadcasts to several receivers. Similarly, if efficiency requires use of the shorthand version of an address for communication between the sender and the receiver, this clearly implies that the sender and the receiver have negotiated this address, and agree to use it over some sequence of messages. Again, this idea makes no sense if communication is isolated in unsequenced messages.

Another problem that is traditionally handled in the context of a sequence of messages is the acknowledgment to the sender that the receiver has correctly received a message. If messages are sequenced, acknowledgment can be very easily done by acknowledging the highest member of the sequence that has been successfully received. If messages bear no relationship to each other, then each must be identified uniquely by the sender, and acknowledged uniquely by the receiver. This increases the complexity and overhead of acknowledgment. However, in most cases where message exchange communication is the appropriate underlying communication model, no acknowledgment mechanism is required of the low-level protocol at all. For example, if a microprocessor system asks the time of day, it is not at all necessary to acknowledge that the query has been successfully received; the receipt of the correct time is sufficient acknowledgment. Similarly, a request for a network address is acknowledged by a return message that contains the desired address. Depending on a low-level acknowledgment message to handle all failures can be dangerous, for it may lead to the practice of assuming that acknowledgment of receipt of a message implies that the message was processed at a high level.

In the broadcast context, it is difficult to formulate a useful definition of acknowledgment that can be supported by a low-level protocol. What does it mean to say that a broadsive. For example, the ARPANET NCP host-to-host protocol [26] initiates a connection using a 56-bit (net, host, socket) identifier for the destination, but then goes through a negotiation so that instead of sending this 56-bit value on subsequent messages, a 32-bit (net, host, link) value can be sent instead. It is not clear whether this conservation of bits is appropriate even in a long-haul network; in a local area network, where bandwidth is inexpensive, it is clearly irrelevant. Other examples of ways in which extra header space can be used to simplify processing include:

1) having a single standard header format with fields in fixed locations, rather than having optional fields or multiple packet types; field extraction at the host can be optimized, reducing processing time;
2) using addresses that directly translate into addresses of queues, buffers, ports, or processes at the receiver without table lookup.

b) Simplified flow control, etc.: The low transmission delay inherent in local area networks, as well as their high data rate, can eliminate the need for complex buffer management, flow control, and network congestion control mechanisms. Consider, for example, flow control: the problem of assuring that messages arrive at the recipient at the rate it can handle, neither too fast, so that its buffers overflow, nor too slow, so that it must wait for the next message after processing the previous one. In a long-haul network, a receiver typically allocates to the transmitter enough buffer space for several messages following the one currently processed by the receiver, so that messages can be placed in transit well before the receiver is ready to process them. Considerable mechanism is required to keep the sender and the receiver properly synchronized under these circumstances. In a local area network, the delay will typically be low enough for a much simpler flow control mechanism to be employed. For example, one can use the very simple strategy of not sending a message until the recipient has explicitly indicated, by a message in the other direction, that it is ready for it. In contrast, a network using communication satellites has such a high transmission delay that very complex predictive flow control algorithms must be used to obtain reasonable data throughput.

It is crucial to understand that other factors may obviate these simplifications. While the data rate and delay characteristics of a local area network can render it essentially instantaneous, its speed cannot eliminate the intrinsic disparity that may exist between the capabilities of two hosts that wish to communicate with each other. These disparities may not show up when the two hosts are communicating through a long-haul network whose characteristics are so constraining that the principal problem is dealing with the restrictions of the network. While protocols for local area networks need not include mechanisms designed to cope with the limitations of the network itself, it is still necessary to design protocols with sufficient generality to cope with disparities between the capabilities of machines wishing to communicate through the network. Such disparities include:

1) mismatch between the rate at which hosts can generate and absorb data;
2) host delay between the time a packet is received and the time it is successfully processed and acknowledged:
3) amount of buffer space available at the sender and the receiver.

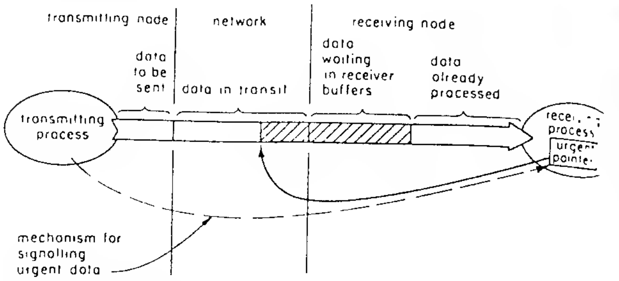


Fig. 9. The urgent pointer mechanism. By transmitting a new, larger value of the urgent pointer, a pointer into the data stream, a sender can indicate the data bufferred in the sender, network, and receiver are holding up data that must be processed quickly. The receiver can then adjust his use of the data stream flow control to process the bufferred data until the urgent data is processed. The shaded area indicates the location of potentially urgent data specified by a particular urgent pointer value.

Further, considerable effort may be required to modify host software to provide a suitable interface to the network. If one were to consider the simple flow control mechanism mentioned earlier, where a message is sent in the reverse direction requesting transmission of each message as it is needed, one would discover that in many cases the scheme was unworkable, not because the network introduced intolerable delays, but because the hosts communicating with each other themselves introduced excessive delay. In a large host with a time-shared operating system, for example, the real time that elapses from the time a message is received, one or more processes are scheduled in response to this message, and that process runs, to the time a message is sent in response, could well run into a large number of milliseconds milliseconds during which the other host is forced to wait.

c) Example of protocol simplification: The low-level protocol initially proposed for the Laboratory for Computer Science Network at MIT is an example of the sort of protocol that results when simplicity of mechanism is a primary design goal. The Data Stream Protocol (DSP) was based on the Transmission Control Protocol (TCP) used in internetworking experiments sponsored by the Defense Advanced Research Projects Agency [27], but evolved from original TCP due to the continuing desire to simplify the protocol features, packet formats, and implementation strategies. Most of these simplifications have subsequently been incorporated into the TCP

One specific example is the mechanism used to signal *interrupts* and other urgent messages that are logically part of the sequence of data in a virtual circuit. The basic model is that the sender occasionally wants to signal the receiver that all data in the stream preceding the signal (buffered somewhere in the network) must be scanned immediately in order to respond promptly to some other important signal. A mechanism is provided whereby a pointer into the data stream is maintained at the receiver, which can be moved, when the sender chooses, to point to a more recently transmitted piece of data. This pointer, called the *urgent pointer*, can be used to indicate the point in the data stream beyond which there is no more urgent data. (See Fig. 9.) The urgent pointer can be implemented in two ways, depending upon the nature of the host receiving the message. In the case of a simple (e.g., microprocessor) host dedicated to a task that processes the incoming stream as it arrives, the host need not process the urgent pointer, since by design, all data, urgent or not, are processed as quickly as possible. In contrast, on a large time-shared host, data need not be processed until either

The basic design principle of the transceiver is that it must present a high impedence to the bus except when it is transmitting and actually driving the bus. This is essential to the operation of the contention bus network; a large number of receivers on the bus must not present impedence lumps or in any way interfere with a transceiver which is actively transmitting.

The receiver must be able to detect and properly receive signals from the most distant point on the bus; in addition, it must be able to detect a colliding signal while its companion transmitter is itself driving the bus. This requirement impacts the choice of an encoding scheme for data transmitted on the bus. A number of data encoding schemes can be used, all of which require that the transmitter be able to place the transmission medium in two distinct states. At first glance, it might seem that *three* states could be used: the quiescent, high-impedence state, to indicate that no transmission is in progress, and two active driver states, for example $+V$ and $-V$. However, with two active driver states, when two or more network nodes attempt to transmit simultaneously, the cable will be driven to different voltage levels at different points. This has two effects. First, it places a severe load on drivers. Second, it makes the detection of a colliding signal more difficult than it needs to be. On the other hand, if the transceiver drives the cable to some voltage to represent one signaling state, and represents the other signaling state by *not* driving the cable, the problem of overloaded drivers is eliminated, and the task of collision detection is greatly simplified. Collision detection is accomplished looking at the bus during the transmitter's quiescent state. Any signal present during that time must come from another transceiver, and constitutes a collision. The transceiver can detect an incoming signal with 20-dB attenuation, which corresponds to about 1 km of the particular cable used.

The transceiver must be able to cope with ground potential differences at the various network hosts. Isolation is accomplished by high-speed optocouplers and an isolated power supply which enables the major circuit elements of the transceiver to be referenced to cable ground, rather than local host ground. Finally, the fault detection, or watchdog circuit examines the output of the driver to guard against transceiver failures which drive the bus and disrupt the network. The signaling states used by the transceiver result in the driver being quiescent approximately 50 percent of the time; if the driver remains on steadily for several bit-times, it is deemed to be faulty, and the fault detector disconnects its power, which, of course, returns the driver to its high-impedence state.

5) Complexity of the Local Network Interface: In its present form, the LNI comprises about 350 TTL SSI and MSI integrated circuits, apportioned as follows:

PDP-11 full-duplex DMA	100
Name table controller	25
Name table cells (8 provided)	90
Network-oriented portion	120
Test and diagnostic	15
Total	350

The count of 120 chips for the network-oriented portion of the LNI, excluding the associative name table, is well within the capabilities of current large-scale integration. As the field of local area networking matures, and standards are arrived at, it is likely that integrated circuit manufacturers will add local area network controllers to their product lines, to take their place alongside other LSI data communication chips which are already available, making high-performance local area network technology available at a very reasonable cost.

V. Protocols for Local Area Networks

As in long-haul networks, local area network protocols can be divided into two basic levels—low-level protocols and high-level protocols. At each level, the characteristics of local networks impact effects on protocol design and functionality.

A. Low-Level Protocols

The term *low-level* protocol identifies the basic protocols used to transport groups of bits through the network with appropriate timeliness and reliability. The low-level protocols are not aware of the meaning of the bits being transported, as distinct from higher level application protocols that use the bits to communicate about remote actions. Two aspects of local area networks have a very strong impact upon the design of low-level protocols. First, the high performance achievable purely through hardware technology enables the simplification of protocols. Second, low-level protocols must be designed to take advantage of and preserve the special capabilities of local networks, so that these capabilities can be utilized, in turn, by higher level application protocols. We will explore these two issues in this section.

1) Simplicity: Local area networks must support a wide variety of hosts, from dedicated microprocessors to large time-sharing systems. The existence of extremely simple hosts (such as microprocessor-based intelligent terminals, or even microprocessor printer controllers) leads to a desire for simple, flexible, low-level protocols that can be economically implemented on small hosts, while not compromising the performance of large hosts. Supporting a variety of hosts also leads to a difficult software production and maintenance problem that can be ameliorated somewhat by having a protocol that is simple to implement for each new kind of host. Although quite a variety of hosts has been attached to long-haul networks such as the ARPANET, the problem of software development has not been too severe, since each individual host in such environments usually has a software maintenance and development staff. In the local area network context where a variety of computers are all maintained by a small programming staff, the arguments for simplicity in protocol design are far stronger in our view.

In a long-haul network, complexity results from strategies that attempt to make as much of the costly network bandwidth as possible available for transport of high-level data. The costs of a local area network are concentrated instead in the host interfaces, the hosts themselves, and their software. Two factors lead to the simplicity of low-level local area network protocols.

a) Unrestricted use of overhead bits: Bandwidth is inexpensive in a local area network; there is little motivation to be concerned with protocol features designed to reduce the size of the header or overhead bits sent with each message. This is in contrast to protocols developed for networks making the more conventional assumption that bandwidth is expenof a message. In a contention bus or contention ring network, the output machine may transmit only when the network is quiet. The "token present" signal is replaced by a "network quiet" signal. In the ring network, the reception control section signals the transmission control section if it detects another token in the midst of its receipt of the message the transmission control section sent; this has its analogue in the collision detection capability of the contention network. In both cases, the LNI must abort transmission of its message and take corrective action. In the ring network this is an error condition, an exception; more than one control token is present in the ring. In the contention network, a collision is an expected event. Both situations can be handled by the LNI reporting the event to host software, which can attempt to restart a token on the ring, in the ring network case, or apply a retransmission backoff algorithm in the contention network case.

A better solution for the contention network is to modify the transmission control section to execute a simple retransmission backoff algorithm in hardware. This requires that the entire message remain accessible to the transmission control section without host intervention. The FIFO buffer cannot be used in this situation; a complete packet buffer which is not erased until the message has been successfully transmitted is an appropriate alternative.

Two features of the ring network LNI's transmission control section are not needed in the contention bus network version: the data repeater which passes bits from the receive side of the LNI to its transmit side when the LNI is not transmitting a message, and the token generator which places a new token or connector onto a quiescent ring. Of course, the connector is a brief sequence of bits, and there is no good motivation to delete it from the beginning of messages transmitted by the contention bus version of the LNI. In fact, retention of the connector at the head of a message results in fewer changes to the input machine of the LNI. It can use its token/connector detector to signal the beginning of an incoming message. Its function remains the same, for the most part; extra connectors detected in the middle of a message indicate a collision, just as they do for the ring network version. However, in the contention bus network, because bits are not repeated from one LNI to another, there is no way to set the match/accept bits for the benefit of the transmitting LNI, and the match/accept field of the message cannot be usëd.

The signal conditioning section of the LNI undergoes an interesting transformation. For a contention ring network, of course, the signal conditioning section remains the same. However, for a contention bus network, the logic levels of the LNI must be converted to appropriate signal levels and waveforms for the coaxial cable of the bus. This is done in a two-step process. First, a cable transceiver is added to the configuration. To minimize impedence mismatches, reflections, etc., the transceiver is located immediately adjacent to the network cable, and is often packaged separately from the LNI.[4] It is connected to the cable either directly, or via a short stub cable attached to the main cable via a tap. Second, since the transceiver is located adjacent to the network bus cable, and the LNI is located next to its host, an appropriate transmission scheme must be selected to span the intervening distance. For distances up to 30 ft or so, "single-ended" drivers and receivers will suffice. For better reliability, greater distances, or both, differential signals over a shielded twisted pair can be used—just as in the transmission media of the ring network itself. So, the signal conditioning section of the original LNI can be modified to interconnect the LNI and the cable transceiver.

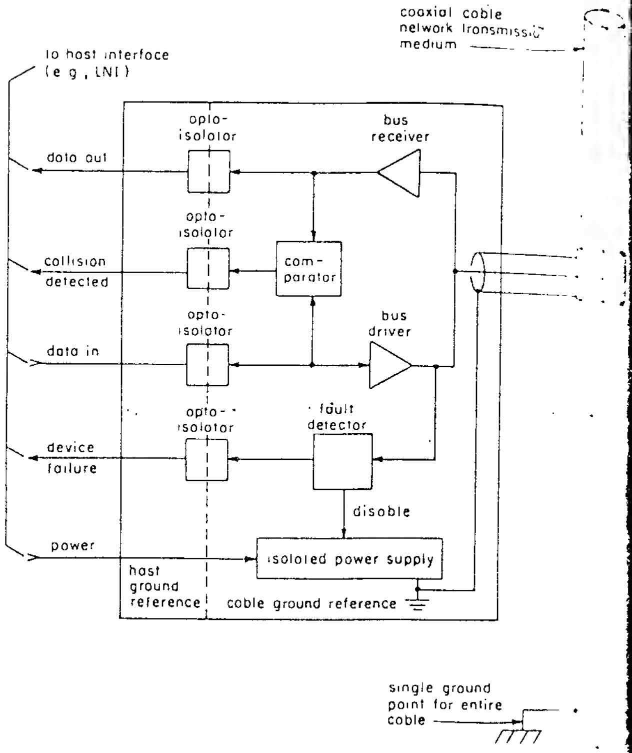


Fig. 8. A typical bus transceiver. The opto-isolators and isolated power supply permit the drivers and receivers to be referenced to cable ground; the cable, in turn, is grounded at only one point along its length, eliminating problems that would result if each transceiver tied the cable to local host ground.

4) The Cable Transceiver: The care taken in the design of a cable transceiver for a contention bus network will strongly influence the overall reliability and performance of the network. Therefore, we conclude our case study by examining a hypothetical contention bus cable transceiver, shown in Fig. 8, that is similar to one designed and built for the CHAOS Network at the MIT Artificial Intelligence Laboratory; it is typical of transceivers built for various contention bus networks.

The cable transceiver performs the following functions:

1) transmission (cable driving);
2) reception;
3) power and ground isolation;
4) collision detection;
5) transceiver fault detection ("watchdog").

The first three of these constitute part of the signal conditioning function described previously.

[4] This has become common practice in local area networking; the networking transmission medium is generally *not* brought into the racks, equipment bays, etc., of a host computer where it would be subject to accidental disconnection and other physical abuse that could disrupt the entire network. Instead, the connection point for a host is designed to be physically stable: a box on the wall, above a false ceiling, etc.